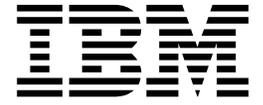


z/OS



Network File System Customization and Operation

z/OS



Network File System Customization and Operation

Note

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 221.

Third Edition, December 2001

This edition applies to Version 1 Release 2 of z/OS™ (5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM® representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
RCF Processing, Department M86/050
5600 Cottle Road
San Jose, CA 95193-0001
United States of America

IBMLink™ from US: STARPUBS at SJEVM5
IBMLink from Canada: STARPUBS at TORIBM
IBM Mail Exchange: USIB3VVD at IBMMAIL
Internet: starpubs@us.ibm.com
World Wide Web: <http://www.storage.ibm.com/software/sms/sms/home.htm>

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1991, 2001. All rights reserved.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	ix
Tables	xi
About this book	xiii
Required product knowledge	xiii
Where to find more information	xiii
Referenced publications	xiv
Using LookAt to look up message explanations	xv
How to Read Syntax Diagrams	xvi
Accessing z/OS books on the internet	xviii
Accessing licensed books on the web	xix
How to send your comments	xix
Related protocol specifications	xx
Summary of changes	xxi
Summary of changes for SC26-7417-02, z/OS Version 1 Release 2	xxi
New information	xxi
Changed information	xxi
Moved information	xxi
Deleted information	xxi
Summary of changes for SC26-7417-01, z/OS Version 1 Release 2	xxii
New information	xxii
Changed information	xxii
Moved information	xxiii
Deleted information	xxiii
Summary of changes for SC26-7417-00, Version 1 Release 1	xxiii
Chapter 1. Introduction	1
Overview	1
Using z/OS UNIX files	2
z/OS UNIX enhancements	2
NFS protocol compliance	2
Using conventional MVS data sets	3
Mounting MVS data sets onto a client mount point	3
Crossing file systems–NFS server	3
Creating conventional MVS data sets	4
Data set serialization and sharing	4
Server control files	4
Attributes data set	4
Exports data set	5
Mount handle data set	5
Log data set	5
Checklist data set	6
Supported clients for the z/OS NFS server	6
Supported servers for the z/OS NFS client	6
WebNFS support	7
NFS version 3 and TCP protocol	7
Native ASCII support	8
Chapter 2. Customization	9
Protecting your data	9
Protecting the server control files	10

Setting up the NFS authorization	10
Protecting the file system on MVS	11
Customized installation security exits	13
Authentication using UNIX style credentials	13
Data conversion	14
Set up conversion environment for Unicode Services	14
Set up NFS servers for multiple TCP/IP stacks	17
Invocation	17
Example procedure to start an NFS server in a multiple server environment	18
User interactions	18
Errors	18
Messages and codes	19
Collecting Network File System usage data	19
Configuring the z/OS NFS client	20
z/OS NFS client considerations	20
Updating MVS system data sets for the client	20
Allocating the z/OS NFS client log data sets	22
Mounting remote file systems	22
Configuring the z/OS NFS server	23
Allocating and modifying the attributes data set	23
Allocating and modifying the exports data set	24
Allocating and modifying the checklist data set	26
Allocating the mount handle data sets	27
ASCII-EBCDIC translation tables	28
Updating MVS system data sets for the server	30
Allocating the z/OS NFS server log data sets	30
Allocating and modifying mapping side file	30
Modifying tcpip.ETC.RPC	30
Installing the client enabling commands	31
Retrieving commands for AIX, Sun Solaris, Sun UNIX, HP/UX	32
Retrieving commands for DEC ULTRIX	35
Retrieving commands for Sun PC-NFS	36
Porting the mvslogin, mvslogout, and showattr commands	37
Porting considerations	39
Chapter 3. Attributes for the z/OS NFS server	43
Attributes used for z/OS UNIX System Services file access	43
Using multipliers	43
Specifying attributes multiple times	43
Data set creation attributes	44
Processing attributes	46
Timeout attributes	50
Retrieve attributes	51
Mapped keyword processing attribute	51
Native ASCII processing attributes	52
Site attributes	54
Chapter 4. Attributes for the z/OS NFS client	61
Client attributes	61
DataCaching attribute	63
Mount processing parameters and installation parameters	64
Chapter 5. Operating the Network File System	65
Starting the z/OS NFS client	65
Stopping the z/OS NFS client	65
Starting the z/OS NFS server	66

Stopping the z/OS NFS server	67
Starting the z/OS NFS NSM and z/OS NFS NLM	68
Starting the z/OS NFS NSM	68
Starting the z/OS NFS NLM	68
Stopping the z/OS NFS NSM and the z/OS NFS NLM	69
Operands of the modify command for the z/OS NFS server	69
Exportfs operand	70
Freeze operand	70
List operand	71
Release operand	72
Status operand	72
Unmount operand	72
Operands of the modify command for diagnosis	73
Flushlog operand	73
Log=msglevel operand	73
Smf operand	74

Chapter 6. Network File System installation-wide exits for the z/OS NFS server

server	75
Requirements for the NFS.	75
Sample link-edit JCL	76
Using storage blocks of the server exits.	76
Login installation-wide exit.	77
Requirements of the login exit	79
Options of the login exit	79
Structure of the login exit message	79
Contents of the login exit parameter list	79
Using the login exit parameter list	80
Request codes to the login exit	80
Return codes from the login exit	80
System initialization routine of the login exit	81
Start of new user session routine of the login exit	81
User login request routine of the login exit	82
User logout request routine of the login exit	82
System termination routine of the login exit	83
File security installation-wide exit	83
Requirements of the file security exit	85
Structure of the file security exit message	85
Contents of the file security exit parameter list	85
Using the file security exit parameter list	87
Request codes to the file security exit	87
Return codes from the file security exit	87
Validate allocate request routine of the file security exit	87
Validate write request routine of the file security exit	88
Validate read request routine of the file security exit	89
Return security permissions routine of the file security exit	89

Chapter 7. Diagnosing and reporting problems

Correcting input error	91
Using keywords to identify the problem	92
Component identification keyword	92
Release level keyword	92
Type-of-failure keyword	93
Service level keyword	95
Log data sets	95
Setting up the z/OS NFS client log data sets	96

Setting up the z/OS NFS server log data set	97
Flushing the z/OS NFS server log	98
Setting up a dump data set for abends	98
Searching the IBM database for APARs and PTFs	98
Contacting the IBM support center.	99
Diagnostic aids	99
First failure data capture	99
Environmental checklist	101
Appendix A. Network File System messages	103
Server messages	103
Client messages	128
Messages from the client platform (AIX)	145
Appendix B. Return codes	149
Appendix C. Sample CHECKLIST data set	153
Appendix D. NFS system server sample attribute table	155
Appendix E. Sample exports data set	167
Appendix F. Sample startup procedures	169
Sample z/OS NFS server startup procedures	169
Sample z/OS NFS client startup procedures.	171
Sample startup procedure for the z/OS NFS NSM	172
Sample startup procedure for the z/OS NFS NLM	173
Appendix G. Retrieving source code for client enabling commands	175
Appendix H. Using the PCNFSD protocol	177
Accessing data with PCNFSD	177
Accessing z/OS UNIX files	177
Starting the PCNFSD server	177
PCNFSD protocols supported	178
Using version 1 of the PCNFSD protocol	178
Using version 2 of the PCNFSD protocol	178
Appendix I. Handling of the file size value	181
Storage of the file size value	181
System-managed PS, VSAM, and PDSE data sets	181
Migrated system-managed data sets	181
Non-system-managed, PDS, and direct access data sets	182
How the file size value is generated.	182
Using fastfilesize to avoid read-for-xize	182
Nofastfilesize	183
Appendix J. Handling of the time stamps.	185
Time stamps for system-managed VSAM and PS data sets	185
Time stamps for non-system-managed PS and DA data sets	185
Storing time stamps	185
Client program requirements	185
Generating time stamps	185
Time stamps for non-system-managed VSAM data sets	186
Time stamps for PDSs and PDSEs	186
Setting time stamps.	187

Appendix K. Sample procedure to generate an SMF report (GFSAPSMF)	189
Appendix L. SMF C and assembler header macros	209
SMF C header macro GFSASSMF	209
SMF assembler header macro GFS AUSMF	215
Notices	221
Programming interface information	222
Trademarks	222
Glossary	223
Index	231

Figures

1.	NFS client-server relationship	1
2.	Sample filesystype parmlib statement	20
3.	Entry for a directory	25
4.	Creating the mount handle data sets	28
5.	Specifying the mount handle data set in the MVS NFS procedure	28
6.	Retrieving the client enabling commands for AIX, Sun Solaris, and HP/UX, UNIX	33
7.	Retrieving the client enabling commands for DEC ULTRIX	35
8.	Retrieving the client enabling commands for Sun PC-NFS version 5.0	37
9.	Sample link-edit JCL for the NFS	76
10.	Determining which login checking routines are used	78
11.	Determining which file security checking routines are used	84
12.	NFS system server sample attribute table	155
13.	Sample exports data set	167
14.	Sample z/OS NFS server startup procedures	169
15.	Sample z/OS NFS client startup procedures	171
16.	Sample startup procedure for the z/OS NFS NSM	172
17.	Sample startup procedure for the z/OS NFS NLM	173
18.	Retrieving source code for client enabling commands	175
19.	Sample procedure to generate an SMF report.	189
20.	SMF C header macro GFSASSMF	209
21.	SMF assembler header macro GFS AUSMF	215

Tables

1. Additional information	xiv
2. View of NFS server capability	7
3. z/OS server processing of a mount request	12
4. z/OS server processing of a file request	13
5. Filedata and delim combinations	21
6. Modifying tcpip.ETC.RPC.	30
7. Files in the prefix.NFSTARB data set to download to clients	31
8. Mvslogin, mvslogout, (or mvslogut), and showattr commands - all platforms	38
9. Attributes - z/OS NFS server	43
10. Data set creation attributes	44
11. Processing attributes	46
12. The mapped keyword and existing keywords	51
13. z/OS NFS clients with non-390 based NFS servers	53
14. NFS servers with non-390 based NFS clients	53
15. Site attributes	55
16. Attributes - z/OS NFS client	61
17. Client attributes	61
18. Mount processing parameters	64
19. Installation parameters.	64
20. List of installation-wide exits.	75
21. Format of login installation-wide exit routine parameter list	79
22. Request codes to the login exit	80
23. Return codes from the login exit	80
24. Codes and fields for system initialization	81
25. Codes and fields for start of new user session	81
26. Codes and fields for user login request.	82
27. Codes and fields for logout request	82
28. Codes and fields for system termination	83
29. Format of the parameter list for the file security installation-wide exit	85
30. Request codes to the file security exit	87
31. Return codes from the file security exit.	87
32. Codes and fields for validate allocate request	87
33. Codes and fields for validate write request	88
34. Codes and fields for validate read request	89
35. Codes and fields for return security permissions	89
36. Summary of type-of-failure keywords	93
37. NFS symptom data	100
38. Dump content and storage areas	100
39. Diagnostic errors and messages.	101
40. Messages - client operating system, NFS server, and NFS client.	103
41. NFS server log data set message format	103
42. NFS client MVS operators console message format	128
43. NFS client log data set message format	129
44. Common variables.	130
45. Parsing error (when reason code is 6E01xxxx)	130
46. Parsing error (when reason code is from 6E0111yy to 6E0129yy)	131
47. Parsing error (when reason code is 6Exxyyy')	131
48. GFSC845I valid parameters	144
49. Externalized return codes defined by the NFS version 2 protocol.	149
50. Externalized return codes defined by the NFS version 3 protocol.	149
51. z/OS NFS version 1 protocol externalized return code mapping to z/OS UNIX	151
52. z/OS NFS return codes between NFS version 3 protocol and z/OS UNIX	151

About this book

This book provides system programmers and operators with information about customizing and operating the z/OS NFS.

This book contains the following information:

- An introduction to the NFS
- Customization information for the z/OS NFS server and client
- Configuration information (such as setting attributes and installing client enabling commands)
- A description of the commands that are available for operating and diagnosing the z/OS NFS server and client
- Information about writing customized exits for login and file security
- Information about using the z/OS NFS Network Lock Manager (NLM) and the z/OS Network Status Manager (NSM)
- Information about diagnosing and reporting problems
- Environmental issues that are encountered in multiple virtual system (MVS)
- Explanations of the messages that are sent to the operator's console or to the z/OS NFS server and client log data sets
- Information about return codes
- A sample exports data set
- A sample sidefile
- A sample checklist
- Sample startup procedures for the z/OS NFS server and client
- Sample startup procedures for the z/OS NFS NLM and the z/OS NFS NSM
- A sample session on how to download source code for client enabling commands
- Source code for a sample system management facilities (SMF) report generator
- A list of related protocol specifications
- Information on handling the file size value
- Information on handling the time stamps
- A sample procedure to generate a SMF report
- Sample SMF C and Assembler header macros
- A list of acronyms and definitions of terms that are used in this book

Required product knowledge

To use this book effectively, you should be familiar with the IBM multiple virtual system (MVS) operating system, the IBM Time Sharing Option (TSO), and their commands. In addition, you should be familiar with System Modification Program/Extended (SMP/E) and the basic concepts of the NFS protocol and networking (Transmission Control Protocol/Internet Protocol (TCP/IP)).

Where to find more information

Where necessary, this book references information in other books, using the shortened version of the book title. For complete titles and order numbers of the books for all products that are part of z/OS, see *z/OS Information Roadmap*.

Referenced publications

The following publications have additional information:

Table 1. Additional information

Publication Title	Order Number
<i>AIX for RISC System/6000 Performance Monitoring and Tuning Guide</i>	SC23-2365
<i>AIX Operating System TCP/IP User's Guide</i>	SC23-2309
<i>AIX Commands Reference for RISC System/6000, Volume 1</i>	GC23-2376
<i>AIX Commands Reference for RISC System/6000, Volume 2</i>	GC23-2366
<i>AIX Commands Reference for RISC System/6000, Volume 3</i>	GC23-2367
<i>AIX Commands Reference for RISC System/6000, Volume 4</i>	GC23-2393
<i>AIX Communications Concepts and Procedures for IBM RISC System/6000</i>	GC23-2203
<i>Character Data Representation Architecture Reference and Registry</i>	SC09-2190
<i>Character Data Representation Architecture Overview</i>	GC09-2207
<i>DFSMS/MVS Online Product Library</i>	LK2T-8731
<i>IBM Online Library: AIX Base Collection Kit</i>	SK2T-2166
<i>IBM Online Library Omnibus Edition: MVS Collection</i>	SK27-0710
<i>IBM Networking Softcopy Collection Kit</i>	SK2T-6012
<i>Introducing IBM's Transmission Control Protocol/Internet Protocol Products for OS/2[®], MVS, VM</i>	GC31-6080
<i>OS/390 Support for Unicode[™]: Using Conversion Services</i>	SC33-7050
<i>SMP/E R8.1 Reference</i>	SC28-1107
<i>TCP/IP: Performance Tuning Guide</i>	SC31-7188
<i>TCP/IP for MVS: Programmer's Reference</i>	SC31-7135
<i>TCP/IP for MVS: Customization and Administration Guide</i>	SC31-7134
<i>TCP/IP for MVS: User's Guide</i>	SC31-7136
<i>z/OS Communications Server IP Configuration Reference</i>	SC31-8776
<i>z/OS DFSMSdfp Advanced Services</i>	SC26-7400
<i>z/OS DFSMSdfp Diagnosis Reference</i>	GY27-7618
<i>z/OS DFSMSdfp Storage Administration Reference</i>	SC26-7402
<i>z/OS DFSMSShsm Diagnosis Guide</i>	LY35-0114
<i>z/OS MVS Programming: Authorized Assembler Services Guide</i>	SA22-7608
<i>z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN ALE-DYN</i>	SA22-7609
<i>z/OS MVS Programming: Authorized Assembler Services Reference ENF-IXG ENF-IXG</i>	SA22-7610
<i>z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU LLA-SDU</i>	SA22-7611
<i>z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO SET-WTO</i>	SA22-7612
<i>z/OS DFSMS Introduction</i>	SC26-7397
<i>z/OS DFSMS Macro Instructions for Data Sets</i>	SC26-7408
<i>z/OS DFSMS Migration</i>	GC26-7398

Table 1. Additional information (continued)

Publication Title	Order Number
<i>z/OS DFSMS: Using Data Sets</i>	SC26-7410
<i>z/OS MVS System Management Facilities (SMF)</i>	SA22-7630
<i>z/OS MVS Installation Exits</i>	SA22-7593
<i>z/OS MVS JCL Reference</i>	SA22-7597
<i>z/OS MVS System Messages, Vol 1 (ABA-ASA)</i>	SA22-7631
<i>z/OS MVS System Messages, Vol 2 (ASB-EWX)</i>	SA22-7632
<i>z/OS MVS System Messages, Vol 3 (GDE-IEB)</i>	SA22-7633
<i>z/OS MVS System Messages, Vol 4 (IEC-IFD)</i>	SA22-7634
<i>z/OS MVS System Messages, Vol 5 (IGD-IZP)</i>	SA22-7635
<i>z/OS MVS Programming: Authorized Assembler Services Guide</i>	SA22-7608
<i>z/OS Network File System Performance Tuning Guide</i>	SC26-7418
<i>z/OS Network File System User's Guide</i>	SC26-7419
<i>z/OS TSO/E User's Guide</i>	SA22-7794
<i>z/OS UNIX System Services File System Interface Reference</i>	SA22-7808
<i>z/OS Information Roadmap</i>	SA22-7500
<i>z/OS UNIX System Services Messages and Codes</i>	SA22-7807
<i>z/OS SecureWay Security Server RACF Security Administrator's Guide</i>	SA22-7683
<i>z/OS SecureWay Security Server RACF System Programmer's Guide</i>	SA22-7681
<i>z/OS SMP/E User's Guide</i>	SA22-7773
<i>z/OS UNIX System Services Programming: Assembler Callable Services Reference</i>	SA22-7803
<i>z/OS UNIX System Services Programming Tools</i>	SA22-7805
<i>z/OS UNIX System Services Command Reference</i>	SA22-7802
<i>z/OS UNIX System Services User's Guide</i>	SA22-7801
<i>z/OS UNIX System Services Planning</i>	GA22-7800

Using LookAt to look up message explanations

LookAt is an online facility that allows you to look up explanations for z/OS messages, system abends, and some codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can access LookAt from the Internet at:

<http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html>

or from anywhere in z/OS where you can access a TSO command line (for example, TSO prompt, ISPF, z/OS UNIX System Services running OMVS).

To find a message explanation on the Internet, go to the LookAt Web site and simply enter the message identifier (for example, IAT1836 or IAT*). You can select a specific release to narrow your search. You can also download code from the *IBM Online Library Omnibus Edition z/OS Collection*, SK2T-6700 and the LookAt Web site so you can access LookAt from a PalmPilot (Palm VIIx suggested).

To use LookAt as a TSO command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO from a disk on your *IBM Online Library Omnibus Edition z/OS Collection*, SK2T-6700 or from the LookAt Web site. To obtain the code from the LookAt Web site, do the following:

1. Go to <http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html>.
2. Click the **News** button.
3. Scroll to **Download LookAt Code for TSO and VM**.
4. Click the ftp link, which will take you to a list of operating systems. Select the appropriate operating system. Then select the appropriate release.
5. Find the **lookat.me** file and follow its detailed instructions.

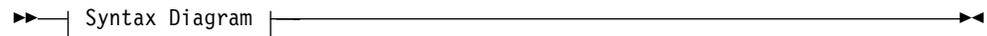
To find a message explanation from a TSO command line, simply enter: **lookat message-id**. LookAt will display the message explanation for the message requested.

Note: Some messages have information in more than one book. For example, IEC192I has routing and descriptor codes listed in *z/OS MVS Routing and Descriptor Codes*. For such messages, LookAt prompts you to choose which book to open.

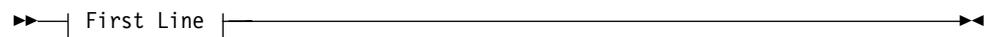
How to Read Syntax Diagrams

Throughout this library, diagrams are used to illustrate the programming syntax. Keyword parameters are parameters that follow the positional parameters. Unless otherwise stated, keyword parameters can be coded in any order. The following list tells you how to interpret the syntax diagrams:

- Read the diagrams from left-to-right, top-to-bottom, following the main path line. Each diagram begins on the left with double arrowheads and ends on the right with two arrowheads facing each other.



- If a diagram is longer than one line, each line to be continued ends with a single arrowhead and the next line begins with a single arrowhead.



- Required keywords and values appear on the main path line. You must code required keywords and values.



If several mutually exclusive required keywords or values exist, they are stacked vertically in alphanumeric order.



- Optional keywords and values appear below the main path line. You can choose not to code optional keywords and values.



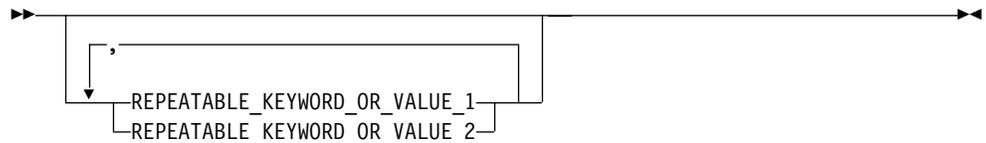
If several mutually exclusive optional keywords or values exist, they are stacked vertically in alphanumeric order below the main path line.



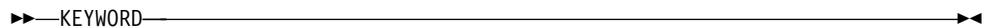
- An arrow returning to the left above a keyword or value on the main path line means that the keyword or value can be repeated. The comma means that each keyword or value must be separated from the next by a comma.



- An arrow returning to the left above a group of keywords or values means more than one can be selected, or a single one can be repeated.



- A word in all uppercase is a keyword or value you must spell exactly as shown. In this example, you must code **KEYWORD**.



If a keyword or value can be abbreviated, the abbreviation is discussed in the text associated with the syntax diagram.

- If a diagram shows a character that is not alphanumeric (such as parentheses, periods, commas, and equal signs), you must code the character as part of the syntax. In this example, you must code **KEYWORD=(001,0.001)**.



- If a diagram shows a blank space, you must code the blank space as part of the syntax. In this example, you must code **KEYWORD=(001 FIXED)**.

▶▶—KEYWORD=(001 FIXED)—▶▶

- Default keywords and values appear above the main path line. If you omit the keyword or value entirely, the default is used.

▶▶—DEFAULT
KEYWORD—▶▶

- A word in all lowercase italics is a *variable*. Where you see a variable in the syntax, you must replace it with one of its allowable names or values, as defined in the text.

▶▶—*variable*—▶▶

- References to syntax notes appear as numbers enclosed in parentheses above the line. Do not code the parentheses or the number.

(1)
▶▶—KEYWORD—▶▶

Notes:

- 1 An example of a syntax note.
- Some diagrams contain *syntax fragments*, which serve to break up diagrams that are too long, too complex, or too repetitious. Syntax fragment names are in mixed case and are shown in the diagram and in the heading of the fragment. The fragment is placed below the main diagram.

▶▶—| Reference to Syntax Fragment |—▶▶

Syntax Fragment:

|—1ST_KEYWORD,2ND_KEYWORD,3RD_KEYWORD—|

Accessing z/OS books on the internet

In addition to making softcopy books available on CD-ROM, IBM provides access to unlicensed z/OS softcopy books on the Internet. To find z/OS books on the Internet, first go to the z/OS home page:

<http://www.ibm.com/servers/eserver/zseries/zos/>

From this Web site, you can link directly to the z/OS softcopy books by selecting the Library icon. You can also link to IBM Direct to order hardcopy books.

Accessing licensed books on the web

z/OS licensed documentation in PDF format is available on the Internet at the IBM Resource Link™ Web site at:

<http://www.ibm.com/servers/resourceLink>

Licensed books are available only to customers with a z/OS license. Access to these books requires an IBM Resource Link Web userid and password, and a key code. With your z/OS order you received a memo that includes this key code.

To obtain your IBM Resource Link Web userid and password, log on to:

<http://www.ibm.com/servers/resourceLink>

To register for access to the z/OS licensed books, follow these steps:

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **User Profiles** located on the left-hand navigation bar.
3. Click on **Access Profile**.
4. Click on **Request Access to Licensed books**.
5. Supply your key code where requested and click on the **Submit** button.

If you supplied the correct key code you will receive confirmation that your request is being processed. After your request is processed you will receive an e-mail confirmation.

Note: You cannot access the z/OS licensed books unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

To access the licensed books, follow these steps::

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **Library**.
3. Click on **zSeries**.
4. Click on **Software**.
5. Click on **z/OS**.
6. Access the licensed book by selecting the appropriate element.

How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or other NFS documentation, you can select one of the following options:

- Visit our home page at:

<http://www.ibm.com/servers/eserver/zseries/zos/nfs/>

From this Web site you can click on **Contact Us** to enter and submit your comments, or click on **Library** to access NFS books.

- Send your comments by e-mail to:
 - IBMLink from US: starpubs@us.ibm.com
 - IBMLink from Canada: STARPUBS at TORIBM
 - IBM Mail Exchange: USIB3VVD at IBMMAIL
 - Internet: starpubs@us.ibm.com

Be sure to include the name of the book, the part number of the book, version and product name, and if applicable, the specific location of the text you are commenting on (for example, a page number or a table number).

- Fill out one of the forms at the back of this book and return it by mail or by giving it to an IBM representative. If the form has been removed, address your comments to IBM Corporation, RCF Processing Department M86/050, 5600 Cottle Road, San Jose, California 95193-0001, U.S.A.

Related protocol specifications

IBM is committed to industry standards. The internet protocol suite is still evolving through Requests for Comments (RFC). New protocols are being designed and implemented by researchers, and are brought to the attention of the internet community in the form of RFCs. Some of these are so useful that they become a recommended protocol. That is, all future implementations for TCP/IP are recommended to implement this particular function or protocol. These become the *de facto* standards on which the TCP/IP protocol suite is built.

The NFS is implemented as a set of RPC procedures that use External Data Representation (XDR) encoding to pass arguments between clients and servers. The NFS is based on the following RFCs:

Internet Protocol	RFC 791, J.B. Postel
NFS: Network File System Version 2 Protocol Specification	RFC 1094, Sun Microsystems, Incorporated
NFS: Network File System Version 3 Protocol Specification	RFC 1813, Sun Microsystems, Incorporated
Open Group Technical Standard Protocols for Interworking: XNFS, Version 3W	Document Number: C702
RPC: Remote Procedure Call Protocol Specification Version 2	RFC 1057, Sun Microsystems Incorporated
RPC: Remote Procedure Call Protocol Specification Version 2	RFC 1831, R. Srinivasan
User Datagram Protocol	RFC 768, J.B. Postel
WebNFS Client Specification	RFC 2054, B. Callaghan
WebNFS Server Specification	RFC 2055, B. Callaghan
XDR: External Data Representation Standard	RFC 1014, Sun Microsystems, Incorporated
XDR: External Data Representation Standard	RFC 1832, R. Srinivasan

For more information about Request for Comments (RFC), see the Internet Engineering Task Force (IETF) home page:

<http://www.ietf.org/>

RFC documentation is available at the Internet Engineering Task Force (IETF) Web site:

<http://www.ietf.org/rfc.html>

Summary of changes

The summary of changes informs you of changes to this book. Revision bars (|) in the left margin of the book indicate changes from the previous edition.

Summary of changes for SC26-7417-02, z/OS Version 1 Release 2

This book contains information that was previously presented in *z/OS Network File System Customization and Operation*, SC26-7417-01. The following sections summarize the changes to that information.

New information

This edition includes the following new information:

- Server information; see “Set up NFS servers for multiple TCP/IP stacks” on page 17.
- New site attribute **hfsftimeout**; see “Site attributes” on page 54.
- Message information; see “Server messages” on page 103.
- Reference documentation; see Table 1 on page xiv.

Changed information

This edition includes the following changed information:

- Customizing Network File System servers; see “Collecting Network File System usage data” on page 19.
- Site attribute values; see “Site attributes” on page 54.
- Server customizing; see “Starting the z/OS NFS server” on page 66.
- Messages including (<PROCNAME>); see “Server messages” on page 103.
- Sample information; see “Appendix C. Sample CHECKLIST data set” on page 153.
- Locating message information; see “Using LookAt to look up message explanations” on page xv.

Moved information

This edition does not include moved information.

Deleted information

This edition deleted the server message (GFSA905I) information from “Server messages” on page 103 and change history information from examples throughout the book.

This book contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Starting with z/OS V1R2, you may notice changes in the style and structure of some content in this book. For example, headings that use uppercase for the first letter of initial words only, and text that has a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our books.

Summary of changes for SC26-7417-01, z/OS Version 1 Release 2

Attention: This was a major revision of the Network File System (NFS) features in OS/390® (5647-A01 (V2R6, V2R7, V2R8, V2R9, V2R10)) and z/OS (5694-A01 (V1R1, V1R2)); it provides information for NFS features that are applicable to these releases.

This book contains information that was previously presented in *z/OS Network File System Customization and Operation*, SC26-7417-00. The following sections summarize the changes to that information.

New information

This edition includes the following new information:

- Network File System (NFS) features; see “Chapter 1. Introduction” on page 1.
- Reason codes; see Table 47 on page 131.
- Mixing file systems; see “Crossing file systems–NFS server” on page 3.
- File access (UNIX System Services); see “Attributes used for z/OS UNIX System Services file access” on page 43.
- 64-bit virtual user buffer support.
- Binary and text support on the same mount point.
- Mapped keyword processing attribute; see “Mapped keyword processing attribute” on page 51.
- Unicode™ Services (conversion); see “Set up conversion environment for Unicode Services” on page 14.
- Native ASCII support (file tagging); see “Native ASCII support” on page 8, and “Considerations for Native ASCII environment support” on page 52.
- Native ASCII attributes; see “Native ASCII processing attributes” on page 52.
- TCP transport, including a new parameter **proto(tcp|udp)**. The proto parameter lets the NFS client select the transport for communication with another NFS server; see “Updating MVS system data sets for the client” on page 20.
- Client data; see “Data conversion” on page 14.
- Communication with the NFS server, including a new parameter **vers(2/3)**; see “PARMLIB updates” on page 20.
- Processing attributes: **mapped**, **rdrverf**, **nordrverf**, **cln_ccsid(n)**, and **srv_ccsid(n)**; see “Considerations for Native ASCII environment support” on page 52, “Processing attributes” on page 46, “Mapped keyword processing attribute” on page 51, and “Native ASCII processing attributes” on page 52.
- A site attribute **readdirtimeout**; see “Site attributes” on page 54.
- A sample NFS.MAPPING file; see “Appendix D. NFS system server sample attribute table” on page 155.
- TCP client support messages and new file tagging support messages; see “Appendix A. Network File System messages” on page 103.
- Message information; see “Server messages” on page 103, and “Client messages” on page 128.
- Client platform (AIX) messages; see “Messages from the client platform (AIX)” on page 145.
- Glossary terms; see “Glossary” on page 223.

Changed information

This edition includes the following changed information:

- Unicode Standard client support; see “Data conversion” on page 14.
- Stopping the client; see “Stopping the z/OS NFS client” on page 65.
- The name change of OpenEdition® to z/OS UNIX® System Services. Occurrences of z/OS OpenEdition have been changed to z/OS UNIX System

Services or its abbreviated name, z/OS USS. OpenEdition may continue to appear in messages, panel text, and other code with z/OS UNIX System Services.

- Enhancement support; see “z/OS UNIX enhancements” on page 2.
- **unmount** operand information about rebuilding control blocks; see “Unmount operand” on page 72.
- **reuse** parameter information; see “Allocating the mount handle data sets” on page 27.
- **nfstasks** and **security** parameter information; see “Site attributes” on page 54.
- **trusted** attribute information; see “Setting up the NFS authorization” on page 10.

Moved information

This edition includes the following moved information:

- Request comments; see “Related protocol specifications” on page xx.
- Client attributes; see “Chapter 4. Attributes for the z/OS NFS client” on page 61.

Deleted information

This edition deletes the following information:

- OS/2 references are removed; however, some OS/2 references may continue to display in titles, example text, and other source files.
- TCP/IP for DOS references are removed; however, TCP/IP and DOS references may continue to appear in titles, example text, and other source files.
- User number (UID) information as a requirement for the NFS server from “Setting up the NFS authorization” on page 10.

This book contains terminology, maintenance, and editorial changes, including changes to improve consistency and retrievability.

Summary of changes for SC26-7417-00, Version 1 Release 1

This book applies to Version 1 Release 1 of z/OS and contains information that was previously presented in *OS/390 Network File System Customization and Operation*, SC26-7253.

Chapter 1. Introduction

This chapter explains the NFS client-server relationship and introduces the IBM Network File System (z/OS NFS). When used to access z/OS UNIX System Services (z/OS UNIX) data, which conforms to portable operating system interface (POSIX) standards, it is similar to other UNIX/AIX® NFS systems.

Overview

A client is a computer or process that requests services on the network. A server is a computer or process that responds to a request for service from a client. A user accesses a service, which allows the use of data or other resources.

Figure 1 illustrates the client-server relationship. The upper right portion of the figure shows the z/OS NFS server. The lower right portion of the figure shows the z/OS NFS client. The left portion of the figure shows various NFS clients and servers which can interact with the z/OS NFS server and client. The center of the figure shows the Transmission Control Protocol/Internet Protocol (TCP/IP) network used to communicate between the clients and servers.

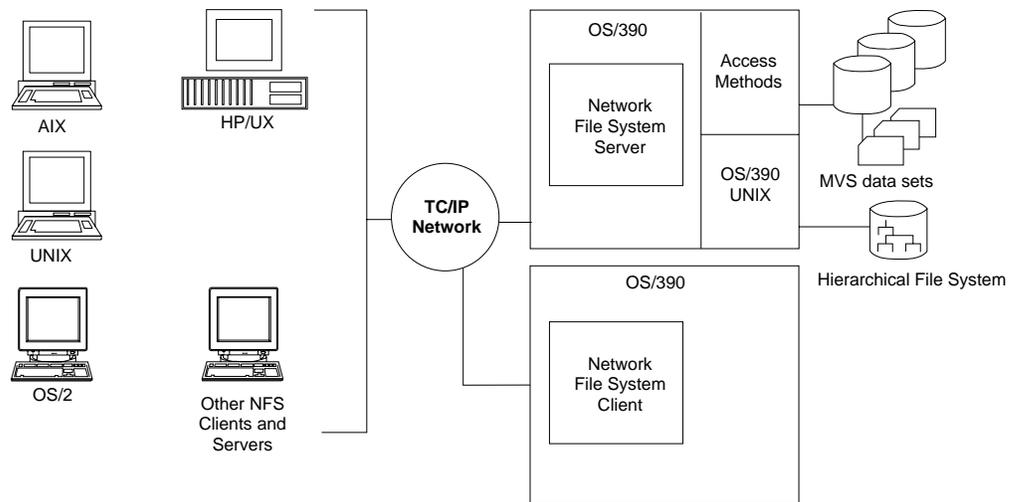


Figure 1. NFS client-server relationship

With the z/OS NFS server, you can remotely access MVS/ESA™ conventional data sets or z/OS UNIX files from workstations, personal computers, and other systems that run client software for the Sun NFS version 2 protocols, the Sun NFS version 3 protocols, and the WebNFS protocols over TCP/IP network.

The z/OS NFS server acts as an intermediary to read, write, create or delete z/OS UNIX files and MVS data sets that are maintained on an MVS host system. The remote MVS data sets or z/OS UNIX files are mounted from the host processor to appear as local directories and files on the client system. This server makes the strengths of an MVS host processor (storage management, high-performance disk storage, security, and centralized data) available to the client platforms.

With the z/OS NFS client you can allow basic sequential access method (BSAM), queued sequential access method (QSAM), virtual storage access method (VSAM), and z/OS UNIX users and applications transparent access to data on systems

which support the Sun NFS version 2 protocols and the Sun NFS version 3 protocols. The remote NFS Server can be an MVS, UNIX, AIX, or other system. The z/OS NFS client is implemented on z/OS UNIX and implements the client portion of the Sun NFS version 2 protocols and the Sun NFS version 3 Protocols.

The NFS uses the communication services provided by TCP/IP, a suite of protocols that includes the remote procedure call (RPC) and External Data Representation (XDR) protocols. RPC allows a program on one machine to start a procedure on another machine, as if the procedure is local. XDR resolves the differences in data representation of different machines.

The NFS, then, can be used for file sharing between platforms and file serving (as a data repository).

If you are using NFS as a file server, the hierarchical file system (HFS) might be a better choice than using conventional MVS data sets, because of its UNIX-like features.

Using z/OS UNIX files

The NFS server enables the client user remote access to z/OS UNIX files from a client workstation.

z/OS UNIX provides a hierarchical file system (HFS) for MVS. The HFS file system is similar to a UNIX file system. All z/OS UNIX files reside in a directory, which in turn is a file in a higher level directory. The highest level directory is called the root directory.

When client users mount files from your server system, you use a common HFS prefix to distinguish z/OS UNIX files from conventional MVS data sets. You see z/OS UNIX files in a standard UNIX format on your workstation, but the files are stored on an MVS host system.

Using the NFS, the client can mount all or part of the z/OS UNIX file system and make it appear as part of your local file system. From there the client user can create, delete, read, write, and treat the host-located files as part of the workstation's own file system. For more information about z/OS UNIX see *z/OS UNIX System Services User's Guide*.

z/OS UNIX enhancements

z/OS UNIX file system support provides these enhancements over conventional MVS data sets:

- Support for hierarchical directories
- File names up to 255 characters in length
- Path names up to 1023 characters in length
- Mixed case names and special characters, except nulls and slash characters, in file and path names
- UNIX-style access permission support
- Group and user ID support at a file level
- Ability to link conventional MVS data sets to a POSIX path name

NFS protocol compliance

The NFS provides full NFS protocol compliance for accessing the HFS file system.

Using conventional MVS data sets

Using the NFS, you can access conventional MVS data sets from a client workstation, personal computer, or any client system using software for the NFS protocol.

In MVS, a file is called a data set. The NFS allows client users to mount conventional MVS data sets from their workstations. It presents the information to them in the form of a UNIX (or AIX) or DOS file, though the information is actually stored on an MVS-owned DASD.

The files for an operating system are organized into a file system. The UNIX and DOS environments use a file system that is a hierarchy of directories. Conventional MVS, in contrast to z/OS UNIX, uses a non-hierarchical file system in which groups of data sets are referred to by specifying a high-level qualifier (HLQ).

The MVS HLQ can include the first (leftmost) qualifier of data sets, or the first and second qualifiers, or the first, second, and third qualifiers, and so on. For example, SMITH is the HLQ for the files named SMITH.TEST.DATA and SMITH.PROJ7.SCHED, while SMITH.TEST is the HLQ of SMITH.TEST.DATA and SMITH.TEST.DOCS.

Mounting MVS data sets onto a client mount point

To access an MVS file system from your client, client users use the **mount** command to create a temporary link (until unmounted) between specific MVS data sets and your UNIX directory (preferably empty) or an unused logical drive on their workstations. The empty UNIX directory or logical drive is called a *mount point*.

Client users use an MVS HLQ in the **mount** command to specify which MVS data sets to mount at a mount point. The MVS data sets beginning with the specified HLQ appears as files under the mount point.

Client users can also perform a mount using a fully qualified data set name or an alias to a user catalog, but not the catalog name itself. Only Integrated Catalog Facility (ICF) cataloged data sets are supported by the z/OS NFS server. Tape data sets and generation data sets are not supported.

Some client platforms support TCP in addition to UDP. Users can choose either TCP or UDP to access the server. The default protocol option depends on the NFS client platform.

Note: When directly mounting on a fully qualified data set name, the server must return the mount size as part of getting the attributes for the mount. This can slow down the completion of the **mount** command.

Crossing file systems—NFS server

Crossing file systems means the NFS client can also potentially be a server, and remote and local mounted file systems can be freely mixed. This leads to some problems when a client travels down the directory tree of a remote file system and reaches the mount point on the server for another remote file system. Allowing the server to follow the second remote mount would require loop detection, server lookup, and user revalidation. When a client does a lookup on a directory on which the server has mounted a file system, the client sees the underlying directory instead of the mounted directory.

The NFS server does not support crossing file systems. For example, if a server has a file system called `/usr` and mounts another file system on `/usr/src`, a client can also mount `/usr`, but the server will not see the mounted version of `/usr/src`. A client could perform remote mounts that match the server's mount points to maintain the server's view. In this example, the client would also have to mount `/usr/src` in addition to `/usr`, even if the mounts are from the same server.

Creating conventional MVS data sets

Client users can create MVS data sets from a client system using the NFS. The default data set creation attributes specified by the system administrator are used to create MVS data sets, unless the user overrides them. These attributes determine how the MVS data sets are structured and where they are stored. Client users can override the default data set creation and processing attributes for a mount point when issuing the **mount** command. In addition, you can override these attributes at file creation time.

Data set serialization and sharing

The z/OS NFS server handles data set serialization and sharing differently, depending on the type of data set:

Physical sequential	The server insures the physical sequential data set read/write integrity by SVC 99 dynamic allocation with exclusive option whenever a physical sequential data set is opened for output. Otherwise, it allocates with share option.
Virtual Storage Access Method (VSAM)	The server dynamically allocates a VSAM data set with share option and allows the VSAM access method to manage data sharing using the shareoptions specified during data set definition.
Partitioned data set extended (PDSE)	The server dynamically allocates a PDSE data set with share option and allows the PDSE functions to manage the serialization of the PDSE data set and its members.
Partitioned data set (PDS)	For read and write, the z/OS NFS server issues ENQ SHR on QNAME=SYSDSN and RNAME=dataset_name (through an SVC 99). For write, the server issues an exclusive ENQ against QNAME=SPFEDIT and RNAME=dataset_name.member_name, in addition to the serialization of resources by SVC 99. For all MVS users who are allocating their data set with exclusive status, this provides write protection. It only provides read integrity for ISPF users.

Server control files

These special files are used by the MVS system administrator to control the z/OS NFS server:

- Attributes data set
- Exports data set
- Mount handle data sets
- Log data sets
- Checklist data set

Attributes data set

The attributes data set contains the settings for the z/OS NFS server. There are three types of attributes stored in this data set:

Data set creation attributes	Used to define the structure of MVS data sets when creating a file (for conventional MVS data sets only)
File processing attributes	Used to control how files are accessed by the client
Site attributes	Used to control z/OS NFS server resources

The system administrator changes the default settings by editing the attributes data set and restarting the server. Client users can override the data set creation and file processing attributes at the command line. For conventional MVS data sets, the client user can specify the data set creation attributes when mounting, creating, or accessing files. The client user can override the file processing attributes when mounting, creating, or accessing files. However, some file processing attributes can only be overridden on a *mount point* basis.

Note: Many of the attributes are valid for conventional MVS files only, and not for z/OS UNIX files. “Attributes used for z/OS UNIX System Services file access” on page 43 gives a complete list of attributes that are valid for z/OS UNIX.

Exports data set

The exports data set can control which client users can mount which MVS data sets. The entries in the exports data set specify which MVS high-level qualifiers or HFS directories can be mounted. The system administrator can use this data set to limit mounts to accredited clients only. It also controls which client users can mount all or part of the z/OS UNIX file system, based on the client machine’s specified Internet Protocol (IP) address. To use the exports data set, the **security** site attribute must be set to either **safexp** or **exports** by the MVS system administrator.

Mount handle data set

The z/OS NFS server maintains a list of the active *mount points* in a pair of files called the mount handle data sets on MVS. The two data sets are used alternately to automatically reestablish the client *mount points* when the server is started. If the file system is not available, the mount point is not reestablished and the mount failure is recorded in the LOG data set.

The z/OS NFS server does the cleanup activity during z/OS NFS server shutdown and daily at the cleanup time specified by the **restimeout** site attribute.

During cleanup time, the z/OS NFS server reads the list and checks all mount points against the retention period specified in the **restimeout** site attribute. If your *mount points* are idle longer than the retention period specified in the **restimeout** site attribute, they are removed. Only the active *mount points* are reconnected.

If a mount handle is removed by the cleanup activity, the client user might receive the “Stale NFS File Handle” message or some other appropriate message. If so, all the client user needs to do is unmount the stale *mount point* and mount it again.

Log data set

The *log data sets* store the messages for the z/OS NFS start-up procedures. This log can be used to identify the users correctable errors or the users problem errors. There are two logs that this information is stored in; the primary log and the secondary log. The primary log is used at start-up until it is filled and then overflows into the secondary log. When the secondary log is full, the primary log will then be overwritten with new error messages. The number of log records is dependent on the number of transactions that the server can handle.

Checklist data set

The checklist data set contains entries for files or directories that are to be exempt from System Authorization Facility (SAF) checking even though SAF or SAFEXP is specified as the security options. This file is only used when SAF or SAFEXP is specified for the particular data type and the **checklist** option is specified as a site attribute. The entries specified here must match a subsequent mount point, or be the parent of a subsequent mount point, to allow SAF checking to be bypassed for everything underneath that mount point. If the entry does not match a subsequent mount point and it is not a parent of any subsequent mount point then it has no effect.

Supported clients for the z/OS NFS server

The following IBM systems are available for NFS client:

- z/OS NFS
- IBM RS/6000® AIX version 4.2.0

NFS client software for other IBM platforms is available from other vendors. You can also access the z/OS NFS server from non-IBM clients that use the NFS version 2 protocol, including:

- DEC stations running DEC ULTRIX version 4.4
- HP 9000 workstations running HP/UX version 10.20
- Sun PC-NFS version 5

You can also access the z/OS NFS server from non-IBM clients that use the NFS version 3 protocol, including:

- Sun workstations running SunOS or Sun Solaris versions 2.5.3
- AIX 4.2.0

Note: This information reflects supported clients as of September 1998. For current information on supported clients, contact IBM.

Supported servers for the z/OS NFS client

The z/OS NFS client supports all servers that implement the server portion of the Sun NFS Version 2 or Version 3 protocols.

A **mount** parameter **vers(x)**, where *x* is either 2 or 3, is provided to make the z/OS NFS client communicate with the server at the specified protocol level. The z/OS NFS client also communicates at the highest protocol level that is supported by the server if no level is specified.

If **no version** is specified and if the server:

- Only supports NFS version 2 protocol, then the z/OS NFS client will use NFS version 2 protocol to communicate
- Supports both the NFS version 2 and 3 protocols, then z/OS NFS client will use NFS version 3 protocol to communicate

If **vers(2)** is specified, then use NFS version 2 protocol to communicate with the server.

If **vers(3)** is specified, then use NFS version 3 protocol to communicate with the server. z/OS NFS client fails the **mount** command if the server does not support NFS version 3 protocol.

WebNFS support

The z/OS NFS server supports the WebNFS protocol. WebNFS specification extends the semantics of NFS versions 2 and 3 protocols to allow clients to obtain file handles without the mount protocols. The z/OS NFS server supports the public filehandle and multi-component **lookup** features as well as other additional requirements as described in RFC 2055. A new keyword, **public**, is added for the system administrator to specify the public paths that the public file handle can access. A public path for conventional MVS data and a public path for HFS data can both be specified. When a **lookup** request comes in from an NFS client and an absolute path name is specified, it will be matched with the public paths to determine which public path it is trying to reference. If a relative path is specified and both HFS and MVS public paths are defined then the **lookup** request will be processed relative to the HFS public path.

The following are restrictions for the WebNFS support provided by the z/OS NFS server in this release.

Export Spanning Pathnames - **lookup** requests, which reference files or directories outside of the exported public path, will result in an error condition.

Symbolic Links - A symbolic link embedded in a multi-component pathname **lookup** request will result in an error condition. However, if the final component is a symbolic link, the server will return the filehandle of the symbolic link and let the client evaluate it. External links, which are special cases of symbolic links, will be handled similarly.

Native Path - Only canonical pathnames will be supported.

NFS version 3 and TCP protocol

The information for NFS version 3 protocol and **proto=tcp** can be found on the *mount* man page on a UNIX client. The NFS client automatically selects the option unless the end-user overrides the option. For example, you can enter this command:

```
unix$ mount -o vers=2,proto=udp mvshost1:smith /mnt
```

The above example shows the preference of NFS version 2 with **udp** protocol, even though the client platform can handle the NFS version 3 and **tcp** protocol.

Users can issue the `rpcinfo -p <hostname>` to show all the RPC programs available on the server. Table 2 illustrates this example.

```
$ rpcinfo -p mvshost1
```

Table 2. View of NFS server capability

program	vers	proto	port	service
100000	2	udp	111	portmapper
100000	2	tcp	111	portmapper
100003	2	udp	2049	nfs

Table 2. View of NFS server capability (continued)

program	vers	proto	port	service
100003	<u>3</u>	udp	<u>2049</u>	<u>nfs</u>
100059	<u>2</u>	udp	<u>2012</u>	
100044	1	udp	2013	
100005	1	udp	2014	mountd
100005	3	udp	2015	mountd
150001	1	udp	2016	pcnfsd
150001	2	udp	2017	pcnfsd
100044	<u>1</u>	<u>tcp</u>	<u>2010</u>	
100059	<u>2</u>	<u>tcp</u>	<u>2011</u>	
100003	<u>2</u>	<u>tcp</u>	<u>2049</u>	<u>nfs</u>
100003	<u>3</u>	<u>tcp</u>	<u>2049</u>	<u>nfs</u>

Note: New information is represented in underlined text.

Native ASCII support

The z/OS NFS client and server support applications running on z/OS V1R2 (and higher) in a "Native ASCII" environment. Applications can operate on files in either EBCDIC or ASCII format as well as other data formats defined with a Coded Character Set Identifier (CCSID). Native ASCII support is provided with a mechanism called file tagging where the file is defined with a tag to identify the CCSID to use for data conversion. File tagging is defined in the appropriate z/OS UNIX System Services (USS) publications. The z/OS NFS client and server provide the necessary support to provide data conversion between different CCSIDs specified for the client and server. The z/OS NFS client *cln_ccsid* and *srv_ccsid* parameters will also be supported by the z/OS NFS server to identify the CCSID to be used in the data conversion. See "Processing attributes" on page 46 for more information about the *cln_ccsid* and *srv_ccsid* parameters.

Chapter 2. Customization

This chapter describes how to configure the NFS and how to make it available to users.

To prepare to configure the z/OS NFS client and server, perform the following tasks:

- Protect your server control files.
- Set up the z/OS NFS authorization.
- Protect the file systems on MVS.
- Use customized security exits.
- Authenticate using UNIX-style credentials.
- Collect z/OS NFS usage data.

To configure the z/OS NFS client, perform the following tasks:

- Note the z/OS NFS client considerations.
- Set up the z/OS NFS client authorization.
- Update the MVS system data sets (PARMLIB, PROCLIB, and DD statement).
- Allocate the primary and secondary z/OS NFS client log data sets.
- Mount the remote file systems.

To configure the z/OS NFS server, perform the following tasks:

- Set up the z/OS NFS server authorization.
- Protect the z/OS NFS server control files.
- Allocate and specify attributes in the attributes data set.
- Allocate and specify entries in the exports data set.
- Allocate the mount handle data sets.
- Update the MVS system data sets (PARMLIB, PROCLIB, and DD statements).
- Allocate the primary and secondary z/OS NFS server log data sets.
- Allocate and modify the checklist data set (if needed).
- Allocate and modify the default mapping side file (if needed).
- Modify tcpip.ETC.RPC.

To install and port the client enabling commands, perform the following tasks:

- Install the **mvslogin**, **showattr**, and **mvslogout** (or **mvslogut**) client enabling commands.
- Port the **mvslogin**, **showattr**, and **mvslogout** (or **mvslogut**) client enabling commands for your client environment.

Note: If a PC client supports PCNFSD and keeps the UID and GID to each mount point base, the client does not need to port the **mvslogin** and **mvslogut** commands. See “Appendix H. Using the PCNFSD protocol” on page 177 for details on PCNFSD support.

Protecting your data

This section covers the following topics:

- Protecting the server control files
- Setting up the NFS authorization
- Protecting the file system

- Using customized installation security exits
- Using UNIX-style credentials

Protecting the server control files

You should protect the server's control data sets from unauthorized access with Resource Authorization Control Facility (RACF®). These data sets are the:

- Attributes file (read by the server at initialization)
- Exports file (read by the server)
- Mount handle data set (read and updated by the server)
- Checklist data set (read by the server)

Setting up the NFS authorization

The following security measures should be addressed when you install the z/OS NFS server and client:

- All programs that come with the z/OS NFS server and client must reside in an APF-authorized program library.
- You need to define the z/OS NFS server and client to RACF and assign the necessary level of authority. You do this by defining a RACF user ID for the z/OS NFS server and client. Because the z/OS NFS server and client are run as started tasks, you also need to define an entry in the RACF-started procedures table which associates the z/OS NFS server and client startup procedure names with the previously defined user IDs. For more information about coding and replacing the RACF-started procedure table, see *z/OS SecureWay Security Server RACF Security Administrator's Guide* and *z/OS SecureWay Security Server RACF System Programmer's Guide*.

The z/OS NFS server can now be set up with the **trusted** attribute as follows:

```

SETROPTS GENERIC(STARTED)
RDEFINE STARTED mvsnfs.* STDATA(USER(mvsnfs) GROUP(SYS1) TRUSTED(YES))
SETR CLASSACT(STARTED)
SETR RACLIST(STARTED)
ADDUSER mvsnfs OMVS(UID(n)) /* where n is any integer */

```

With trusted authority, the NFS server can perform the following tasks:

- Reconstruct the mount points (from the active mount handle data set) upon startup
- Handle mount requests from client prior to user login
- Handle `ls` or `nfsdir` list commands prior to user login
- Be a trusted user during normal operation

During actual remote client file access, the z/OS NFS server first RACROUTEs the remote client's user ID to determine if the remote client is authorized to access the data set. If the remote client is authorized, the z/OS NFS server switches to its own user ID, which has trusted authority, to access the data set.

- You need to define a z/OS UNIX segment for the z/OS NFS client in the RACF user profile as UID(0).
- You need to define a z/OS UNIX segment for the z/OS NFS NLM and the z/OS NFS NSM in the RACF user profile as UID(0).

For TCP/IP security information, see *TCP/IP for MVS: Customization and Administration Guide*.

For z/OS UNIX security information, see *z/OS UNIX System Services Planning*.

Protecting the file system on MVS

You can select the level of protection you want for different types of data access. A different protection level can be specified for the three data access types: MVS™ (legacy data), HFS data, and public data (data accessed using the public file handle). The security site attribute is used to protect the three different types of data accesses. The format of the keyword is **security**(*mvs,hfs,public*) where *mvs,hfs,public* are place holders for the four different security options: NONE, SAF, SAFEXP, EXPORTS. See “Site attributes” on page 54 for syntax rules. The z/OS NFS server can be configured to handle security in the following ways:

- None
- Exports list checking
- System Authorization Facility (SAF) checking
- Customized installation security exit
- System Authorization Facility (SAF) checking with checklist processing (to bypass SAF for files and directories under selected mount points)
- A combination of these approaches

Note: The UNIX permission checking against the z/OS UNIX hierarchical file system might appear to be inconsistent if the definition of UID and GID is not consistent throughout the domain of the network.

Unrestricted data access — **security(none)**

If you do not want to restrict data access, you can use the **security(none)** site attribute. Neither exports list checking nor SAF checking is done. Client users can access MVS files without an MVS user ID and without using the **mvslogin** command. They simply mount the MVS files that they want to access and unmount when they are finished. For HFS files, the UNIX permission bits are checked before access is granted to the client user. The UNIX permission checking is based on the UID from the RPC request.

Note: If UID or GID from the RPC request is zero, it will be mapped to 65534 (nobody) before the UNIX permission checking is performed.

Exports list checking — **security(exports)**

When you specify **security(exports)** in the attributes data set, the NFS checks the client IP address against the exports list, which is generated from the exports data set, to determine whether or not a mount is to be granted. The NFS also checks the requested directory (or high-level qualifier) to be mounted. For HFS data, it also checks the UNIX permission bits before granting file access to a client user. The UNIX permission checking is based on the UID from the RPC request. Since SAF checking is not done, client users do not need to use the **mvslogin** command.

Note: If UID or GID from the RPC request is zero, it will be mapped to 65534 (nobody) before the UNIX permission is performed.

For more information about the exports data set, see “Allocating and modifying the exports data set” on page 24.

SAF checking — **security(saf)**

When you specify **security(saf)** in the attributes data set, the Network File System uses Resource Access Control Facility (RACF) or an equivalent product to control access to MVS data sets. All RACF requests from the server are made through SAF. SAF directs control to RACF, or an equivalent security product, if it is active.

The server uses SAF to validate the MVS user ID and password supplied by the client user. It also uses SAF to validate that the client user is allowed to access the data set on MVS. A RACF user ID must be defined for each client user that requires access to the server.

For HFS data, z/OS UNIX checks the UNIX permission bits before granting file access to a client user. The UNIX permission checking is based on the z/OS UNIX UID obtained by the **mvslogin** process, not the UID passed in by the client on the RPC request. For users accessing HFS, their RACF user ID must have an z/OS UNIX segment defined in the RACF profile.

Both SAF and exports list checking — security(safexp)

When you specify **security(safexp)** in the attributes data set, the NFS checks for both RACF authorization and exports list authorization before granting a client user access to z/OS UNIX data. For HFS data, z/OS UNIX checks the UNIX permission bits before granting file access to a client user. The UNIX permission checking is based on the z/OS UNIX UID obtained by the **mvslogin** process, not the UID passed in by the client on the RPC request. This is the most restrictive means of limiting DASD access. It requires client users to use the **mvslogin** command.

For more information about the exports data set, see “Allocating and modifying the exports data set” on page 24.

SAF checking with checklist processing

When you specify **security(saf)** or **security(safexp)** with the checklist site attribute, the Network File System performs SAF as described in “Both SAF and exports list checking — security(safexp)”. The only exception to this is that it will not check the files and directories which are underneath the mount points that either match the mount point or the children of the mount points that are in the checklist data set. This allows users that do not have MVS userids access to data without having to do a MVSLOGIN while still maintaining SAF checking for data that requires MVS userids. For more information, see “Appendix C. Sample CHECKLIST data set” on page 153.

Exporting a file system

A system administrator issues the **mount** command to an NFS server and makes a remote file system available to the user. The z/OS server keeps a list of file systems and associated access restrictions in an export file. It then compares incoming mount requests to the entries in the file. If a match is found in the export file and the client is authorized for access, then the file system is successfully mounted.

Table 3 shows server processing of a mount request.

Table 3. z/OS server processing of a mount request

Security Option	Export File	HFS file	MVS dataset
none	not required	no checking, exported	no checking exported
saf	not required	no checking exported	no checking exported
exports	required	checking export file	checking export file
safexp	required	checking export file	checking export file

Note: MVSLOGIN is not required for NFS mount request.

Authorizing file operations

After the file system is mounted, the user perform the normal file operations on the NFS-mounted remote file system. z/OS NFS server adds the MVS SAF checking in addition to the UNIX file permissions check. For SAF security, the user must perform **mvslogin** to authenticate to MVS before attempting to do any file access. Otherwise, the user will be denied access because of insufficient authorization.

Note: MVS conventional data set does not support UNIX permission bits in the file attribute structure. By disabling the SAF security, there is no authorization checking for file operation to MVS conventional data set. The UNIX permission bits checking is still performed for HFS file operations when the SAF security is disabled.

Table 4 shows server processing of a file request.

Table 4. z/OS server processing of a file request

Security Option	MVSLOGIN	HFS file	MVS dataset
none	not required	check file permission bits	no checking
saf	required***	SAF check***	SAF check***
exports	not required	check file permission bits	no checking
safexp	required***	SAF check***	SAF check***

Note: z/OS UNIX segment must be defined for HFS file operation. (***)This does not apply when checklist requirements are satisfied.)

Customized installation security exits

You can write installation-wide exits or replaceable modules that customize Network File System security processing, by using product-sensitive programming interfaces provided by the server. Depending on how you code the exit, client users could be required to use the **mvslogin** command even for the **security(none)** and **security(exports)** attributes.

For more information about customizing your installation's security exits see "Login installation-wide exit" on page 77 and "File security installation-wide exit" on page 83.

Authentication using UNIX style credentials

Authentication is the process of verifying the identity of a client system. This ensures that one client system cannot masquerade as another client system (perhaps with a different set of privileges). Client systems are identified by a set of credentials and authenticated with verification information passed in messages sent to server systems. There are several different conventions for exchanging authentication information in the NFS protocol, including these credentials:

- Null credentials
- UNIX-style credentials
- DES-style credentials
- Other, user-written, credentials

UNIX-style credentials are the only variety supported by the z/OS NFS. The z/OS NFS client utilizes z/OS UNIX-socket-enabled RPCs to communicate with remote z/OS NFS servers over a TCP/IP network. The credential includes the user ID (UID), group ID (GID), and a list of GIDs the user belongs to.

Data conversion

The z/OS NFS client supports data conversion defined by the universal character encoding standard known as the Unicode Standard on z/OS V1R2 (and above) when reading data from a remote NFS server or writing data to a remote z/OS NFS server. The Unicode Standard offers character conversion as well as basic case conversion. Within character conversion, characters are converted from one coded character set identifier (CCSID) to another. CCSID information is obtained from the **cln_ccsid** and **srv_ccsid** parameters.

On z/OS releases below V1R2, the z/OS NFS client supports data conversion defined by the Character Data Representation Architecture (CDRA) when reading data from a remote NFS server or writing data to a remote NFS server. CDRA characters are converted from one CCSID to another. CCSID information is obtained from the **cln_ccsid** and **srv_ccsid** parameters. See *Character Data Representation Architecture Reference and Registry* and *Character Data Representation Architecture Overview* for additional information.

Only single byte to single byte data conversion is supported. For example, if a client file has a CCSID of 437 and a server file has a CCSID of 297, data conversion will occur between USA ASCII format (CCSID 437) and French EBCDIC format (CCSID 297). Single byte to multiple byte conversion (including double byte character set (DBCS)) is not supported and will result in NFS terminating with an error message.

The **cln_ccsid**, **srv_ccsid**, and **xlat** parameters identify whether data conversion is performed, and how data conversion is done. The **cln_ccsid**, **srv_ccsid**, and **xlat** parameters are supported by the z/OS NFS client installation parameter and **tso mount** command. The parameters on a **tso mount** command override the parameters specified as a z/OS NFS client installation parameter.

Set up conversion environment for Unicode Services

The z/OS client or server uses Unicode Services to support data conversion on files in either EBCDIC or ASCII formats as well as other data formats that are defined with a CCSID. Once the Unicode Services have been installed, a conversion environment has to be set up to create a conversion image that is loaded into storage. The conversion image contains conversion tables that define the data conversion allowed between CCSIDs. See *Program Directory for OS/390 V2 R8/R9/R10 Support for Unicode™*, G110–9760, and *Support for Unicode™: Using Conversion Services*, SC33-7050, for more information.

The Unicode Services set up requirements information is described in the following items.

1. Follow the *Support for Unicode™: Using Conversion Services* to obtain SCUNJCL, SCUNMSG, SCUNREXX, SCUNTBL data sets.
2. Change the CUNJIMS1 member of *prefix*.SCUNJCL:

```

//UNIMSG1 JOB (JOB0001),'UNI-IN',NOTIFY=&SYSUID,
//  MSGCLASS=X,MSGLEVEL=(1,1),TIME=60,CLASS=A,
//  REGION=0M
//*****
//*
//* LICENSED MATERIALS - PROPERTY OF IBM
//*
//* 5647-A01
//*
//* (C) COPYRIGHT IBM CORP. 2000
//*
//* STATUS = HUNI2A0
//*
//*****
//DEFINENU EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SMSFMC DD  UNIT=SYSDA,VOL=SER=VM5U16,DISP=OLD
//SYSIN DD  *
        DEFINE CLUSTER (NAME(prefix.V280) -
                        VOLUMES(VM5U16) -
                        CYL(1 1) -
                        SHAREOPTIONS(1,3) -
                        LINEAR) -
        DATA (NAME(prefix.V280.MMS.DATA))
/*

```

and submit it.

3. Change the CUNJIMS2 member of *prefix*.SCUNJCL:

```

//UNIMSG2 JOB (JOB0002),'UNI-IN',NOTIFY=&SYSUID,
//  MSGCLASS=X,MSGLEVEL=(1,1),TIME=60,CLASS=A,
//  REGION=0M
//*****
//*
//* LICENSED MATERIALS - PROPERTY OF IBM
//*
//* 5647-A01
//*
//* (C) COPYRIGHT IBM CORP. 2000
//*
//* STATUS = HUNI2A0
//*
//*****
//COPYENU EXEC PGM=IEBCOPY
//SYSPRINT DD  SYSOUT=*
//IN DD  DISP=SHR,DSN=prefix.SCUNMSG
//OUT DD  DISP=(NEW,PASS),
//  UNIT=SYSALLDA,SPACE=(CYL,(10,10,10)),
//  DCB=(RECFM=VB,LRECL=259,BLKSIZE=27998)
//SYSIN DD  *
        COPY OUTDD=OUT,INDD=IN
        SELECT MEMBER=(CUNIIENU)
/*
//*
//*
//COMPENU EXEC PGM=CNLCCLPLR,PARM=(ENU,N)
//SYSPRINT DD  SYSOUT=*
//SYSUT1 DD  DISP=(OLD,DELETE),DSN=*.COPYENU.OUT
//SYSUT2 DD  DISP=OLD,DSN=prefix.V280.MMS.DATA
//*
//*****
//*

```

and submit it.

4. Create the MMSLST01 member in SYS1.PARMLIB:

```

/*-----*/
/* MEMBER = MMSLST01 */
/* */
/* PARMLIB MEMBER FOR MVS MESSAGE SERVICE */
/* */
/* DESCRIPTION = THIS PARMLIB MEMBER IS ACCESSED BY SPECIFYING */
/* MMS(01) ON THE INIT STATEMENT OF THE CONSOLXX */
/* MEMBER OF SYS1.PARMLIB OR BY ISSUING THE */
/* OPERATOR SET MMS=01 COMMAND. */
/* */
/* */
/*-----*/
DEFAULTS LANGCODE(ENU)
LANGUAGE LANGCODE(ENU) DSN(IBMUSER.UN.V280.MMS.DATA) CONFIG(CNLENU00)
NAME(ENGLISH)

```

- Issue the **SET MMS=01** command from the system console:

```
SET MMS=01
```

- Change the CUNJIUTL member of *prefix*.SCUNJCL:

```

//UNIUTL JOB (JOB0003),'UNI-IN',NOTIFY=&SYSUID,
// MSGCLASS=X,MSGLEVEL=(1,1),TIME=60,CLASS=A,
// REGION=0M
//*****
//*
//* LICENSED MATERIALS - PROPERTY OF IBM
//*
//* 5647-A01
//*
//* (C) COPYRIGHT IBM CORP. 2000
//*
//* STATUS = HUNI2A0
//*
//*****
//*
//* IMAGE GENERATOR
//*
//*****
//CUNMIUTL EXEC PGM=CUNMIUTL
//SYSPRINT DD SYSOUT=*
//TABIN DD DISP=SHR,DSN=prefix.SCUNTBL
//SYSIMG DD DSN=prefix.V280.IMAGES(CUNIMG01),DISP=SHR
//SYSIN DD *

/*-----*/
* INPUT STATEMENTS FOR THE IMAGE GENERATOR *
/*-----*/

CASE NORMAL; /* ENABLE TOUPPER AND TOLOWER */

CONVERSION 1047,850; /* EBCDIC -> ASCII */
CONVERSION 850,1047; /* ASCII -> EBCDIC */
CONVERSION 1047,819; /* EBCDIC -> ASCII */
CONVERSION 819,1047; /* ASCII -> EBCDIC */
CONVERSION 277,819; /* EBCDIC -> ASCII */
CONVERSION 819,277; /* ASCII -> EBCDIC */

/*

```

- Copy CUNIMG01 conversion image from *prefix*.V280.IMAGES to SYS1.PARMLIB.
- Create member CUNUNI01 in SYS1.PARMLIB:

```

***** Top of Data *****
IMAGE CUNIMG01;
REALSTORAGE 0;
***** Bottom of Data *****

```

9. Include the "UNI=01, " line into the IEASYSxx member of SYS1.PARMLIB.
10. Re-IPL the system.

If Unicode Services is set up correctly, the "D UNI,ALL" command from system console will issue:

```
00- 05.01.28 SYSTEM1          d uni,all
    05.01.29 SYSTEM1          CUN3000I 05.01.28 UNI DISPLAY 205
      ENVIRONMENT: CREATED    05/14/2001 AT 00.21.17
                    MODIFIED   05/14/2001 AT 00.21.19
                    IMAGE CREATED 05/08/2001 AT 01.50.36
      SERVICE: CUNMCNV      CUNMCASE
      STORAGE: ACTIVE      162 PAGES
                    LIMIT    524287 PAGES
      CASECONV: NORMAL
      CONVERSION: 01047-00850-          00850-01047-
                  01047-00819-          00819-01047-
                  00277-00819-          00819-00277-
                  00939-00819-          00819-00939-
```

The **D MMS,ALL** command from the system console will issue:

```
00- 05.38.33 SYSTEM1          d mms, all
    05.38.37 SYSTEM1          CNLS026I 05.38.36 MMS DISPLAY 580
      PARMLIB MEMBER = MMSLST01
      LAST REFRESH WAS AT 05:37 ON 05/15/2001
      CODE CONFIG OBJECT
      ENU  CNLENU00  prefix.V280.MMS.DATA
```

Set up NFS servers for multiple TCP/IP stacks

This enhancement exploits the ability of the z/OS Communication Server to configure up to 8 TCP/IP stacks simultaneously. One NFS server will be allowed to run on each TCP/IP stack. All NFS servers have their own IP-address and work independently of each other with each connecting to a specific transport provider. So each NFS server will use its own unique set of data sets for mount handle database, error log, site attributes, startup procedures, and exports list.

The client works with any NFS server as an independent host. At startup, the client selects an NFS server using the servers IP-address or HOST-NAME on the **mount** parameter. On shutdown of one of the NFS servers, all the clients connected to that server will be forced to make new connections with another NFS server and to repeat the startup procedures such as mvsllogins, mount connections, etc.

Multiple NFS server support provides an environment on z/OS where applications can have system flexibility by running a NFS server on each LPAR of one z/OS system. This lets you, for example, have a production and test NFS server run on one z/OS system. The use of multiple NFS servers also provides the ability to define separate security-schemes, to separate workload on different NFS servers, and use separate attribute definitions.

Each IP-address will have its own NFS server and Lockd/Statd to handle incoming requests, but the support for Lockd/Statd will be available.

Invocation

To run multiple NFS servers on one z/OS system, it is necessary to have a corresponding number of active TCP/IP stacks. Each NFS server and TCP/IP startup procedures for each TCP/IP stack should have different names. Each NFS server startup procedure needs to have the following change:

1. Add the **envvar** parameter with the `_BPXK_SETIBMOPT_TRANSPORT` environment variable to point to the TCP/IP startup procedure.
2. SYSTCPD DD statement to point to its TCP/IP stack profile.

See the example in “Example procedure to start an NFS server in a multiple server environment”.

Example procedure to start an NFS server in a multiple server environment

The following contains a sample procedure to start an NFS server in a multiple server environment.

```
//MVS NFS PROC MODULE=GFSAMAIN,PARMS='INFO',
//          NFSPRFX=MVS NFS,TCPIP=TCPIP.OS390R10
//GFSAMAIN EXEC PGM=&MODULE,
//          REGION=0M,
//          TIME=1440,
//*      Use environment variable _BPXK_SETIBMOPT_TRANSPORT to set
//*      affinity for a specific TCP/IP stack. TCPIPR10 is name of
//*      start procedure for a selected TCP/IP stack.
//*
//          PARM=('ENVAR("_BPXK_SETIBMOPT_TRANSPORT=TCPIPR10")',
//              '/&PARMS')
//*
//*      Define dataset name of selected TCP/IP stack.
//SYSTCPD DD DISP=SHR,DSN=&TCPIP..TCPIP.DATA
//STEPLIB DD DISP=SHR,DSN=USER1.LOADLIB1
//SYSPRINT DD SYSOUT=*
//*
//*SYSDUMP DD DISP=SHR,DSN=&NFSPRFX..SYSDUMP
//OUTPUT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//*
//* Define dataset names which will be used by current NFS server
//NFSLOG1 DD DISP=SHR,DSN=&NFSPRFX..DFSLAB11.LOG11
//NFSLOG2 DD DISP=SHR,DSN=&NFSPRFX..DFSLAB11.LOG21
//NFSATTR DD DISP=SHR,DSN=&NFSPRFX..CNTL(NFSATTRV)
//EXPORTS DD DISP=SHR,DSN=&NFSPRFX..CNTL(EXPORT1)
//FHDBASE DD DISP=SHR,DSN=&NFSPRFX..DFSLAB11.FHDBASE1
//FHDBASE2 DD DISP=SHR,DSN=&NFSPRFX..DFSLAB11.FHDBASE2
```

User interactions

For more information about configuring multiple TCP/IP stacks, see *z/OS UNIX System Services Planning* and *z/OS Communications Server IP Configuration Reference*.

Console Operators will need to distinguish between different NFS server console messages received from multiple NFS servers on one z/OS system.

System administrators can use the new support to define separate security-schemes, separate workload on different NFS servers, and use separate attribute definitions.

Errors

Existing NFS Server console messages display the start procedure name for a specific NFS server.

Messages and codes

Each NFS Server console message displays the NFS server start procedure. For example:

```
GFS320I(<PROCNAME>) NETWORK FILE SYSTEM SERVER INITIALIZATION FAILED: text
```

Where (<PROCNAME>) is the name of the NFS server start procedure.

Collecting Network File System usage data

The z/OS NFS client does not produce any System Management Facilities (SMF) records. However, it does provide the accounting information to z/OS UNIX for SMF recording. z/OS in turn provides the SMF recording services for all physical file systems (PFSs).

The z/OS NFS server does not produce hierarchical file system (HFS) SMF records. However, z/OS UNIX provides the SMF recording services for all physical file systems (PFSs).

You can use the SMF records that the z/OS NFS server produces to keep track of how MVS conventional data sets are accessed, and how long each Network File System user session lasts. The z/OS NFS server writes the following SMF records:

- Record type-42 subtype 7
This record, written when a file times out, provides the Network File System file usage statistics.
- Record type-42 subtype 8
This record, written when a client user logs out of NFS, provides the Network File System user session statistics.

You can control the SMF data collection in the following ways:

- You can use the **smf** site attribute, described in “Site attributes” on page 54, to determine which, if any, SMF statistics are to be collected.
- You can use the SMF=ON or SMF=OFF operand of the **modify** command. See “Smf operand” on page 74 for a description of this command, which turns SMF data collection on and off.
- You can generate an SMF report. See “Appendix K. Sample procedure to generate an SMF report (GFSAPSMF)” on page 189 for the SMF report sample routine, GFSAPSMF, that can be found in the NFSSAMP library.
- You can use the SMF C and Assembler header macros. See “Appendix L. SMF C and assembler header macros” on page 209 for copies of the C header macro, GFSASSMF, and the Assembler header macro GFSASMFM. Both header macros contain the mapping for SMF records and can be found in the NFSMAC library.
- Check the SMF setting in the system in SYS1.PARMLIB(SMFPRMnn) for the SMF record type and subtype, where nn is determined by IEASYSmm and the operator command (SET SMF=nn). The Network File System uses the SMF type 42 record, subtypes 7 and 8. You specify SMF=nn so the system picks the member of SMFPRM with suffix nn.

You can display the SMF settings with the D SMF,0 operator command.

The SMF write macro, SMFWTM, is used to write the SMF records to the SMF data set. When the server starts, the SMF option is disabled. Therefore, the operator needs to explicitly enable the SMF collection.

For more information about SMF see *z/OS MVS System Management Facilities (SMF)*.

Configuring the z/OS NFS client

This section describes how to configure the z/OS NFS client.

z/OS NFS client considerations

During z/OS UNIX file system initialization, the z/OS NFS client is started and run in the logical file system (LFS) colony address space. The **filesystype** parmlib statement for the z/OS NFS client must be present in the SYS1.PARMLIB(BPXPRMxx) parmlib member in order to start the z/OS NFS client. For more information about z/OS UNIX file system reference see *z/OS UNIX System Services File System Interface Reference*.

Updating MVS system data sets for the client

To accommodate the z/OS NFS client you must update MVS system data sets PARMLIB, PROCLIB, and the DD statement.

PARMLIB updates

Add the data set defined in the GFSCPROC STEPLIB containing the z/OS NFS client library to the system's APF authorization list (IEAAPFxx). A sample cataloged procedure named GFSCPROC is provided as a member of the sample library NFSSAMP, see "Sample z/OS NFS client startup procedures" on page 171.

Add the **filesystype** parmlib statement shown in Figure 2 to the z/OS UNIX parmlib member (BPXPRMxx):

```
FILESYSTYPE
TYPE(NFS)
ENTRYPOINT(GFSCINIT)
PARM('installation parms')
ASNAME(proc_name)
```

Figure 2. Sample filesystype parmlib statement

The name in the TYPE operand must be *NFS*, otherwise the utility program *nfsstat* fails.

The operand ENTRYPOINT(GFSCINIT) specifies the entry point for the z/OS NFS client initialization.

The operand PARM('installation parms') specifies the installation parameters for the z/OS NFS client.

The operand ASNAME(*proc_name*) specifies the procedure name in SYS1.PROCLIB that is used by z/OS UNIX to start the address space in which the z/OS NFS client is initialized.

Note: The *proc_name* is also used for the name of the address space.

For data integrity and data isolation among different PFSs, the z/OS NFS client is required to start in a separate and standalone colony address space. To start the NFS client in a separate and standalone colony address space, a unique *proc_name* must be used.

BSAM, QSAM, and VSAM ESDS access to remote files: BSAM, QSAM, and VSAM ESDS applications can access files stored on remote z/OS NFS servers through the z/OS NFS client. This will allow existing MVS application programs access to data on other systems using BSAM, QSAM, and VSAM ESDS interfaces. The BSAM, QSAM, and VSAM ESDS access methods assume that all text files are EBCDIC. When using these access methods, the **delim** parameter indicates whether the remote files contain text or binary data. Text data consists of records that are separated by a delimiter. If the **delim** parameter is not binary, the EBCDIC text delimiter is used by the access methods when processing the remote files. The **delim** parameter is supported on the z/OS NFS client installation parameter and **tso mount** command.

All the remote file objects under the same mount point have the same **delim** value. The **delim** parameter cannot be set on a file basis under the mount point. The **delim** parameter in the **tso mount** command overrides the **delim** parameter specified in z/OS NFS client installation parameter. However, you can override the **delim** parameter on the **tso mount** command with the **filedata** parameter on a JCL DD statement, SVC 99, or TSO **allocate** command. The **filedata** parameter can be either text or binary.

For BSAM, QSAM, and VSAM ESDS applications accessing files stored on remote z/OS NFS servers, the z/OS NFS client provides data conversion when the **xlat=Y** parameter is specified according to the **cln_ccsid** and **srv_ccsid** settings. When **xlat=N**, the z/OS NFS client will not perform data conversion. The **filedata** parameter on a JCL DD statement is also used to specify if the data consists of text records separated by delimiters or if the data is binary and does not contain record delimiters. To avoid undesirable data conversions, care should be taken to insure the specification of the **xlat** and **delim** parameters are not in conflict with the data type specified by the **filedata** parameter on a JCL DD statement. The **filedata** and **delim** parameters only affect BSAM, QSAM, and VSAM ESDS data access and have no affect on the z/OS NFS client data conversion. The z/OS NFS client data conversion is only controlled by the **xlat** parameter. The significance of different **filedata** and **delim** combinations are as follows (**Note:** In each case insure the z/OS NFS client **xlat=Y**, **cln_ccsid**, and **srv_ccsid** parameter settings are correct for the **filedata** and **delim** combination):

Table 5. *Filedata and delim combinations*

Parameter	Description
FILEDATA=TEXT, Delim=notBINARY	Means that the data is to be accessed as text. The access method appends a record delimiter on output and expects delimiters on input. The delimiter used is that specified on the delim parameter.
FILEDATA=TEXT, Delim=BINARY	Means that the data is to be accessed as text. The access method appends a record delimiter on output and expects delimiters on input. The delimiter used is the default of the EBCDIC new line character (x'15') since the delim parameter does not specify a valid text delimiter.

Table 5. Filedata and delim combinations (continued)

Parameter	Description
FILEDATA= BINARY, <i>Delim</i> =notBINARY	Means that the data is to be accessed as binary. The access method does not append record delimiters on output, does not recognize record delimiters on input, and it treats all characters as data on input.
FILEDATA= BINARY, <i>Delim</i> =BINARY	Means that the data is to be accessed as binary. The access method does not append record delimiters on output, does not recognize record delimiters on input, and it treats all characters as data on input.
FILEDATA= not specified, <i>Delim</i> =specified	Means that the data is to be accessed according to the value specified in the <i>Delim</i> parameter.
FILEDATA= not specified, <i>Delim</i> =not specified	Means that the data is to be accessed as binary. The access method does not append record delimiters on output, does not recognize record delimiters on input, and it treats all characters as data on input.

The z/OS NFS client also provides UNIX authentication for security and provides the z/OS UNIX client's UID, GID, and a list of the GIDs from the z/OS UNIX client's groups to the remote NFS server for authorization checking. When the remote NFS server is the z/OS NFS server, the **mvslogin** command can be used to provide additional security checking through RACF authentication. MVS application programs which require access to data on remote z/OS systems may be required to perform an **mvslogin**.

For information about BSAM, QSAM, and VSAM ESDS applications access to HFS or remote files and their restrictions, see *z/OS DFSMS: Using Data Sets*.

PROCLIB updates

Add the procedure name, *proc_name*, specified in the **ASNAME**(*proc_name*) operand to the system PROCLIB.

A sample cataloged procedure named GFSCPROC is provided as a member of the sample library NFSSAMP, see "Sample z/OS NFS client startup procedures" on page 171.

Modify the MVSNFSC procedure and place it in your system PROCLIB. Add the DD statements:

```
NFSCMSG1 as the DD for the primary log data set
NFSCMSG2 as the DD for the secondary log data set
SYSxDUMP as the DD for the SYSxDUMP data set ('x' = U or M)
```

Allocating the z/OS NFS client log data sets

For information about allocating the z/OS NFS client primary and secondary log data sets, see "Log data sets" on page 95.

Mounting remote file systems

z/OS UNIX does not support z/OS NFS mounts in the SYS1.PARMLIB member statement. You can use the z/OS UNIX **automount** facility (*/etc/rc* shell scripts

support) or the **tso mount** command to make a connection between a mount point on your local MVS HFS file system and one or more files on a remote MVS, AIX, UNIX, OS/390, z/OS, or other file system. The remote file system can be mounted using the **tso mount** command only after the z/OS UNIX file system initialization is complete. The **tso mount** command can only be used by a MVS superuser. For additional information about the **tso mount** command, when used with the z/OS NFS client, see *z/OS Network File System User's Guide*.

When using the **automount** facility of UNIX System Services MVS, the remote file system is mounted on its first data access attempt if it is not already mounted.

When the **automount** facility is used to manage remote NFS mount points, the z/OS NFS user could experience ESTALE/EIO errors if the automounter unmounts the accessed mount point when the time limits specified by the **automount duration** and **delay** parameters have been exceeded. The **automount** default, DURATION=NOLIMIT, disables the DURATION timeout period. The DELAY for unmounting file systems should be longer than the time z/OS NFS clients are likely to keep z/OS NFS mounts to the files and directories active. For more information about the z/OS UNIX **automount** facility (*/etc/rc* shell scripts support) see *z/OS UNIX System Services Planning* and *z/OS UNIX System Services File System Interface Reference*.

The remote file system must be mounted on the z/OS UNIX file system prior to any reference is made to the remote data. Once mounted, the remote file system can be treated as an extension of the local z/OS UNIX file system.

Configuring the z/OS NFS server

This section describes how to configure the z/OS NFS server.

Allocating and modifying the attributes data set

To allocate and modify the attributes data set, perform the following tasks:

1. Allocate a fixed-block partitioned data set or a fixed-block sequential data set with a record length of 80.
2. Copy the sample member GFSAPATT from the *prefix.NFSSAMP* data set into the allocated attributes data set.
3. Modify the attributes to suit your installation. You can specify three sets of attributes within the attributes data set:
 - Data set creation attributes.
 - Processing attributes.
 - Site attributes.

Notes:

1. Client users can override the processing and data set creation attributes (for their own sessions), but not the site attributes.
2. The attributes data set specified on the NFSATTR DD statement of the *mvsnfs* startup procedure is read during server startup processing. Further changes to this data set do not take effect until the server is restarted. Also, whenever any attributes are changed, all the previous existing mount points have to be unmounted and mounted again (using the **umount** and **mount** client commands) if you want the mount points to pick up the new attributes.

Specify this attributes data set for the NFSATTR DD statement in the MVS NFS cataloged procedure.

Attribute statement syntax

- You can continue a line in the attributes data set by placing a “\” or “+” at the end of the line.
- A “#,” anywhere in the data set indicates a comment that extends to the end of the line (unless the **altsym** keyword is used in the **start** command or the server startup procedure; if **altsym** is used, a “;” indicates a comment).
- If you specify more than one attribute on a line, separate the attributes with a comma and a space.

Allocating and modifying the exports data set

To allocate and modify the exports data set, perform the following tasks:

1. Allocate a fixed block partitioned data set or a fixed block sequential data set with record length of 80.
2. Copy the sample member GFSAPEXP from the *prefix.NFSSAMP* data set into the allocated data set.
3. Modify the sample exports data set to suit your installation. “Appendix E. Sample exports data set” on page 167 shows the sample exports data set.
4. Specify this exports data set for the EXPORTS DD statement in the MVS NFS cataloged procedure.

The exports data set contains entries for directories that can be exported to clients. It is used by the server to determine which data sets’ high-level qualifiers or HFS directories can be mounted by a client in a read or write mode.

Note: You cannot specify exporting a “parent directory” or a subdirectory of an exported directory. For example, if you specify DIR1 in the exports data set, DIR1 and all its subdirectories are exported. You cannot specify any subdirectories under DIR1 in the exports data set.

When the modified exports data set takes effect

The exports data set specified in the exports DD statement of the MVS NFS startup procedure is read during server startup processing. Changes to this file do not take effect for new mount points until the **exportfs** operand of the **modify** command is completed or the server is restarted. Changes to the file do not affect the active mount points until the mount points are unmounted and remounted.

When the **exportfs** operand of the **modify** command is issued, any errors found in the file are sent to the system log, the entire exports list is not refreshed, and processing continues.

When the server is started, any errors found in the file are sent to the system log, and the server stops once the entire exports data set has been read.

Using the directory statement

Use the *directory* statements in the exports data set to limit access of directories to specified client workstations.

- Entries can be up to 4096 characters long. Directory names must follow MVS naming conventions, unless z/OS UNIX is used.
- Lines can be continued by placing a backslash (\) or a plus sign (+) at the end of the line.
- A “#” anywhere in the data set indicates a comment that extends to the end of the line (unless the **altsym** keyword is used in the **start** command or the server startup procedure; if the **altsym** keyword is used, the “;” indicates a comment).

- Spaces can not be used in the keywords.
- The parameters to the right of *directory* are optional. Except for *ro* and *rw*, the parameters can be combined. If they are combined, only the first parameter is preceded by “-”. For example:

```
user1.test -access=rs60001:sun1:sun2,ro
```

An entry for a directory is specified in Figure 3:

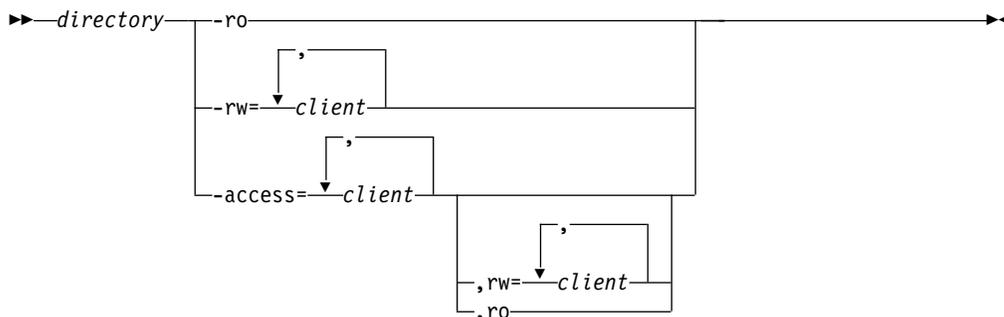


Figure 3. Entry for a directory

directory	MVS high-level qualifier, data set name, or alias to a user catalog; the name must conform to MVS data set naming conventions unless OpenEdition MVS is used (for an HFS directory entry, you need to use the hfs prefix that is used at your site).
-ro	Export the directory as read only. If not specified, the directory is exported as read/write.
-rw=client[:client...]	The directory is exported as read/write to specified <i>clients</i> , and read-only to everyone else. Use a colon (:) to separate client names.
-access=client[:client...] [[,rw=client[:client...]] [,ro]]	<p>Gives access only to <i>clients</i> listed.</p> <p>If neither <i>rw</i> nor <i>ro</i> is specified for the <i>-access</i> parameter, then the clients listed have read/write access and the rest of the clients have no access.</p> <p>If the <i>rw</i> parameter is specified for the <i>-access</i> parameter, the associated clients have read/write access to the directory, and the clients specified in the access list but not in the <i>rw</i> list has read-only access.</p> <p>If the <i>ro</i> parameter is specified for the <i>-access</i> parameter, the clients in the access list have read-only access to the directory, and the rest of the clients have no access.</p> <p>Use a colon (:) to separate client names.</p>

Note: If no options are specified, the default value allows any client to mount the given directory with read/write access.

Following are examples of entries in an exports data set:

```

mvsnfs      -ro      # give read-only access
              # to all clients
              #
theresa.text      # give read/write
              # access to all clients
              #
robert.mixds  -rw=fsrs001:fslab004:fslab007 #
              # give read/write access
              # to the clients named
              # fsrs001, fslab004 and
              # fslab007, and give
              # read-only access to
              # all other clients
              #
/hfs/newproductdirectory -rw=johnson # give read/write access to
              # this HFS directory to the
              # client named johnson;
              # give read-only access to
              # all others
              #
barbara.pds   -access=fsrs001:fslab007 #
              # give read/write access
              # only to clients named
              # fsrs001and fslab007
              #
daniel.pds2   -access=fslab004,ro # give read-only access
              # only to the client
              # named fslab004
              #
virginia.vsam -access=fslab004:fslab007,rw=fslab004 #
              # give read-only
              # access only to the
              # client named fslab007,
              # and give read/write
              # access to fslab004.

```

Notes:

1. If your installation cannot use the “#” as a comment delimiter, see “Starting the z/OS NFS server” on page 66.
2. The keywords **ro** and **rw** are mutually exclusive.
3. The ability to write (that is, **rw** specified or **access** specified without other parameters) implies read access also.
4. If **access** and **rw** are specified together, the client names in the rw list are logically or’ed with the access list to determine the total list of clients with read access.
5. Multiple lines can be used in the exports data set for a given directory to merge the access list and the rw list. However, similar clauses (for example, an access followed by an access) completely replace any previous specification. If **ro** is specified for a data set on one line and a further line specifies **rw** for that data set, the **rw** undoes the **ro** specified earlier. Similarly, a line with null options completely undoes all previous specifications for that directory, giving read/write access to all clients.

Allocating and modifying the checklist data set

To allocate and modify the checklist data set, perform the following tasks:

1. Allocate a fixed block partitioned data set or a fixed block sequential data set with record length of 80.
2. Copy the sample member GFSAPCHK from the *prefix.NFSSAMP* data set into the allocated data set.

3. Modify the sample checklist data set to suit your data security requirements. “Appendix C. Sample CHECKLIST data set” on page 153 shows the sample checklist data set.
4. Specify this checklist data set for the CHKLIST DD statement in the MVS NFS catalogued procedure.

The checklist data set contains entries for prefixes (directories) or file names which should match subsequent mount points or be the parent of the subsequent mount point to allow SAF checking to be bypassed for everything underneath that mount point.

The checklist data set specified in the CHKLIST DD statement of the MVS NFS startup procedure is read during server startup processing. Changes to this file do not take effect for new and active mount points until the server is restarted.

Note: Proper care must be used to protect this data set since it can be used to bypass SAF checking for selected files and directories.

Allocating the mount handle data sets

The mount handle data sets are used to record active mounts during server operation and allow clients to stay mounted when the server is shut down and restarted. The Network File System alternates between two data sets to record this information; only one data set is used at a time, and it is switched at either shutdown or at resource cleanup timeout.

To create the mount handle data sets, perform the following tasks:

1. Allocate two empty VSAM KSDS data sets with the following attributes:
 - Starting with offset 0, the first 16 bytes in the record are the prime key field.
 - The maximum record length of the mount handle data set is 1600 bytes, and the average record length is 400 bytes.

```
DEFINE CLUSTER (NAME(mount_handle_data_set_name) -
               VOL(vsam_volume_name) -
               CYL(1 1) -
               INDEXED -
               REUSE -
               KEYS(16 0) -
               SHAREOPTIONS(3 3) -
               RECSZ(400 1600))
```

Figure 4 on page 28 shows sample JCL, GFSAMHDJ, on how to create a mount handle data set.

2. Specify these two data sets to the FHDBASE DD statement and FHDBASE2 DD statement in the MVS NFS procedure (see Figure 5 on page 28).
3. The server switches data sets after resource cleanup has run.
4. Resource cleanup is done at Network File System shutdown and resource cleanup timeout.

```

//MVSNFS JOB,
//  MSGCLASS=A,MSGLEVEL=(1,1),TIME=30,REGION=12M,CLASS=A
//*
//* CREATE A KSDS DATASET
//*
//DEFINE1 EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DEFINE CLUSTER (NAME(MVSNFS.FHDBASE) -
    VOL(SYS320) -
    CYL(1 1) -
    INDEXED -
    REUSE -
    KEYS(16 0) -
    SHAREOPTIONS(3 3) -
    RECSZ(400 1600))
  LISTC ENT(MVSNFS.FHDBASE) ALL
/*
//*
//* CREATE THE SECOND KSDS DATASET
//*
//DEFINE2 EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DEFINE CLUSTER (NAME(MVSNFS.FHDBASE2) -
    VOL(yyyyyy) -
    CYL(1 1) -
    INDEXED -
    REUSE -
    KEYS(16 0) -
    SHAREOPTIONS(3 3) -
    RECSZ(400 1600))
  LISTC ENT(MVSNFS.FHDBASE2) ALL
/*
//*
```

Figure 4. Creating the mount handle data sets

Figure 5 shows how to specify the mount handle data set on the FHDBASE DD statement and FHDBASE2 DD statement on the MVSNFS procedure.

```

//* FHDBASE AND FHDBASE2 ARE
//* THE MOUNT HANDLE DATA SETS.
//* THEY NEED TO BE PREALLOCATED
//* AS EMPTY VSAM KSDS DATA SETS.
//* THEY WILL BE USED ALTERNATELY.
//* SAMPLE JCL CAN BE FOUND IN HLQ.NFSSAMP(GFSAMHDJ).
//*
//FHDBASE DD DISP=SHR,DSN=MVSNFS.FHDBASE
//FHDBASE2 DD DISP=SHR,DSN=MVSNFS.FHDBASE2
```

Figure 5. Specifying the mount handle data set in the MVSNFS procedure

Note: Delete and allocate the mount handle data sets before running any new versions of the NFS. If an old mount handle data set is used, the server issues a message and shuts down.

ASCII-EBCDIC translation tables

For text processing mode, data is converted between EBCDIC and ASCII. No double byte character set (DBCS) or multiple byte character set (MBCS) forms of data are converted.

Customizing the translation table

You can customize the translation table for the Network File System using the processing attribute **xlat**(*member_name*). The parameter (*member_name*) is the member name of a PDS or PDSE containing the customized translation table. This attribute can be specified either in the mount operation or in the attribute file. It can be specified on a file operation but is ignored, only the mount or the **xlat** value takes effect.

If the processing attribute, **xlat**, is not specified in the attribute file, the Network File System internal translation table is used as the installation default translation table. When the **xlat**(*member_name*) processing attribute is specified in the attribute file, this customized translation table becomes the installation default translation table. The Network File System internal translation table is derived from EBCDIC code page 0037 and ISO 8859-(ASCII). RPC arguments, such as filenames, are always translated by the installation default translation table. Data shipped with RPCs are translated by the mount specified translation table, if any. Otherwise, they are also translated by the installation default translation table.

A mount request with processing attribute, **xlat**, specified overrides the installation default translation table.

When accessing z/OS UNIX files, you must specify the OEMVS311 translation table or your customized translation table either in the mount request or in the default attributes. The OEMVS311 table translates ASCII (ISO 8859-1) to and from EBCDIC (1047 - z/OS UNIX). TCP/IP for MVS version 3.1 provides the OEMVS311 table. This table translates the UNIX line terminator (lf) to the z/OS UNIX line terminator (nl).

See *TCP/IP for MVS: Customization and Administration Guide* for more information about creating and customizing your own translation tables.

Enabling the xlat processing attribute

To enable the **xlat**(*member_name*) processing attribute:

- A DD statement, NFSXLAT, is required in the Network File System startup proc:

```
//NFSXLAT      DD      DSN=dataset_name,DSP=SHR
```

where **dataset_name** is the name of a PDS or PDSE whose member contains the customized translation table.

- A PDS or PDSE, **dataset_name**, is created by the CONXLAT utility whose member contains the customized translation table.

Notes:

1. See *TCP/IP for MVS: Customization and Administration Guide*, "Using Translation Tables," for more information about creating and customizing your own translation tables.
2. You can edit or modify the translation table from your own or from a member in the `tcpip.SEZATCPX` data set and then use CONVLAT utility to convert the source table into binary format. The CONVLAT utility can take a PDS or PDSE as input, and its output data set can be physical sequential, PDS or PDSE.
3. The Network File System only supports PDS and PDSE. A sequential data set must be copied to either a PDS or PDSE member.
4. The Network File System does not support the translation for multiple-byte character sets.

5. Sample steps for creating *xlat* member:
 - a. Run the TCPIP CONVXLAT utility to create a physical sequential (PS) data set with DSORG=PS, RECFM=F, LRECL=256, BLKSIZE=256;

```
"convxlat" 'tcpip.sezactcpx(standard)' 'hlq.xlat.output'
```

- b. Allocate a PDS data set with DSORG=P0, RECFM=F, LRECL=256, BLKSIZE=256; copy the CONVXLAT output data set as a member in the PDS data set
 - c. Allocate the *xlat* member in the z/OS NFS startup procedure.

Updating MVS system data sets for the server

Update the following MVS system data sets to accommodate the z/OS NFS server:

- PARMLIB Updates

Add the data set defined in the GFSAPROC STEPLIB containing the z/OS NFS server library to the system's APF authorization list (IEAAPFxx). A sample cataloged procedure named GFSAPROC is provided as a member of the sample library NFSSAMP, see "Sample z/OS NFS server startup procedures" on page 169.

- PROCLIB Updates

A sample cataloged procedure named GFSAPROC is provided as a member of the sample library NFSSAMP, see "Sample z/OS NFS server startup procedures" on page 169.

Modify the MVS NFS procedure and place it in your system PROCLIB. Add the DD statements:

```
EXPORTS as the DD for the exports data set
NFSATTR as the DD for the attributes data set
FHDBASE and FHDBASE2 as the DD for the mount handle data set
NFSXLAT as the DD to enable the xlat processing attribute
NFSLOG1 as the DD for the primary log data set
NFSLOG2 as the DD for the secondary log data set
SYSxDUMP as the DD for the SYSxDUMP data set ('x' = U or M)
CHKLIST as the DD for the checklist data set (if needed)
```

Allocating the z/OS NFS server log data sets

For information about allocating the z/OS NFS server primary and secondary log data sets, see "Log data sets" on page 95.

Allocating and modifying mapping side file

For information about allocating and modifying mapping side files, see "Appendix D. NFS system server sample attribute table" on page 155.

Modifying tcpip.ETC.RPC

Add the entries in Table 6 to the tcpip.ETC.RPC file for the services provided by the z/OS NFS server:

Table 6. Modifying tcpip.ETC.RPC

Service	Number	Description
nfsd	100003	Network File System daemon
mountd	100005	Mount daemon
mvs mount	100044	MVSmount daemon (for mvslogin, mvslogout)
showattrd	100059	showattr daemon

Table 6. Modifying tcpip.ETC.RPC (continued)

Service	Number	Description
pcnfsd	150001	pcnfs daemon
nlm	100021	network lock manager
nsm	100024	network status monitor

Installing the client enabling commands

To enable client users to access the MVS system and to display system attributes, you must install the **mvlogin**, **mvlogout** (or **mvlogout** for DOS platforms) and **showattr** commands on the client workstations. For some client machines, you might need to modify the code to port these commands so they run on your client machine. See “Porting the mvlogin, mvlogout, and showattr commands” on page 37. Before you install the commands, make sure that TCP/IP and File Transfer Protocol (FTP) are running both on MVS and on the client.

Note: The z/OS NFS client utilities, including **mvlogin**, **mvlogout**, and **showattr**, are installed when the z/OS NFS client and TCP/IP are installed. The target library NFSCUTIL is a DDDEF to an existing HFS directory (/usr/lpp/NFS/IBM) and will contain the client commands for the z/OS NFS client after installation. There is no need to port the z/OS NFS client utilities as you would for the remote NFS clients which use the z/OS NFS server.

Follow these installation procedures:

1. Delete any previous versions of the **mvlogin**, **showattr**, and **mvlogout** (or **mvlogout**) commands and their source code from your client workstation.
2. Retrieve the **mvlogin**, **mvlogout** (or **mvlogout**), and **showattr** commands (in tarbin file format for AIX or UNIX, in executable file format for DOS, or in source code format for any platform) from the *prefix*.NFSTARB data set, where *prefix* is an installation-specified variable. Use FTP with a binary transfer to send the tarbin or executable files to a client workstation for UNIX, AIX, or DOS (no character conversion should be made). Use FTP with text transfer to send the source code of the three commands to a client workstation for any platform.
3. Use the **tar** utility to extract the files only if the client uses AIX or UNIX.
4. Compile the source code only if it is not in executable code format. (You might need to modify the code for your specific client machine.)
5. Make executable code versions of the commands available to all clients.

Notes:

1. We recommend placing these commands on a LAN server (possibly in */usr/local/bin*) that is available to many workstations, rather than installing them on each client workstation.
2. The name of the **mvlogout** command for DOS is **mvlogout**.

Table 7 is a list of all files stored in *prefix*.NFSTARB data set related to the server's client commands (**mvlogin**, **mvlogout** or **mvlogout**, and **showattr**):

Table 7. Files in the *prefix*.NFSTARB data set to download to clients

File Name	Download as:	Description	Client Environment
<i>prefix</i> .NFSTARB(GFSAWAIX)	client.tarbin (or any name)	Binary file	AIX, UNIX, HP/UX

Table 7. Files in the prefix.NFSTARB data set to download to clients (continued)

File Name	Download as:	Description	Client Environment
<i>prefix.NFSTARB</i> (GFSAWDIN)	mvslogin.exe	Binary files; executable code	DOS (using Sun PC-NFS Version 5.0)
<i>prefix.NFSTARB</i> (GFSAWDOU)	mvslogut.exe		
<i>prefix.NFSTARB</i> (GFSAWDSH)	showattr.exe		
Source Code for Commands			
<i>prefix.NFSTARB</i> (GFSAWMNT)	gfsawmnt.h	C header files	All ^{1,2}
<i>prefix.NFSTARB</i> (GFSAWSHO)	gfsawsho.h		
<i>prefix.NFSTARB</i> (GFSAWXAD)	gfsawaxd.c	C modules	All ^{1,2}
<i>prefix.NFSTARB</i> (GFSAWCLT)	gfsawclt.c		
<i>prefix.NFSTARB</i> (GFSAWLIN)	gfsawlin.c		
<i>prefix.NFSTARB</i> (GFSAWLOU)	gfsawlou.c		
<i>prefix.NFSTARB</i> (GFSAWMCL)	gfsawmcl.c		
<i>prefix.NFSTARB</i> (GFSAWMOU)	gfsawmou.c		
<i>prefix.NFSTARB</i> (GFSAWSHA)	gfsawsha.c		
<i>prefix.NFSTARB</i> (GFSAWDLI)	gfsawdli.lnk		
<i>prefix.NFSTARB</i> (GFSAWDLO)	gfsawdlo.lnk		
<i>prefix.NFSTARB</i> (GFSAWDLS)	gfsawdls.lnk		
<i>prefix.NFSTARB</i> (GFSAWMAK)	makefile		All ^{1,2}

Notes:

1. For AIX or UNIX, you do not need to download every individual file if you download the GFSAWAIX file.
2. For DOS you only need to download the executable commands.

The following sample client screens show how to retrieve (for workstations and PCs) and create (for workstations) the **mvslogin**, **mvslogout** or **mvslogut**, and **showattr** client commands for the following platforms:

- AIX, Sun Solaris Versions 2.3 and 2.5.1, Sun UNIX, HP/UX V9.01; see page 32.
- DEC ULTRIX Version 4.4; see page 35.
- Sun PC/NFS Version 5.0; see page 36.

“Appendix G. Retrieving source code for client enabling commands” on page 175 shows how to retrieve the necessary source code to install the client commands on any platform except for an AIX or UNIX workstation.

Retrieving commands for AIX, Sun Solaris, Sun UNIX, HP/UX

Figure 6 on page 33 shows how to retrieve the necessary files to install the client commands on AIX (for RS/6000), HP/UX, or Sun UNIX workstations. **mvshost1** is the name of the MVS host, and **smith** is an MVS user ID.

```

$ ftp mvshost1
Connected to mvshost1
220-FTPSERVE at MVSHOST1, 01:44:24 on 6/02/94
220 Connection closes if idle for more than 5 minutes.
Name (mvshost1:w42dept): smith
<Press ENTER key>
331 Send password please.
Password: password
230 smith is logged on.
ftp> bin
200 Representation type is IMAGE.
ftp> get 'prefix.nfstarb(gfsawaix)' client.tarbin
200 Port request OK.
125 Sending data set PREFIX.NFSTARB(GFSAWAIX)
250 Transfer completed successfully.
local: client.tarbin remote: 'prefix.nfstarb(gfsawaix)'
213504 bytes received in 2.4 seconds (87 Kbytes/s)
ftp> quit
221 Quit command received. Goodbye.
$
$ mkdir mvsnfs.client
$ cd mvsnfs.client
$ tar -xvf ../client.tarbin
tar: record size = 20 blocks
x ./makefile, 8234 bytes, 17 tape blocks
x ./gfsawaxd.c, 13474 bytes, 27 tape blocks
:
$
$ touch *.*
$ ftp mvshost1
Connected to mvshost1

```

Figure 6. Retrieving the client enabling commands for AIX, Sun Solaris, and HP/UX, UNIX (Part 1 of 2)

```

220-FTPSERVE at MVSHOST1, 01:44:24 on 6/02/94
220 Connection closes if idle for more than 5 minutes.
Name (mvshost1:w42dept): smith
<Press ENTER key>
331 Send password please.
Password: password
230 smith is logged on.
ftp> bin
200 Representation type is IMAGE.
ftp> get 'prefix.nfstarb(gfsawaix)' client.tarbin
200 Port request OK.
125 Sending data set PREFIX.NFSTARB(GFSAWAIX)
250 Transfer completed successfully.
local: client.tarbin remote: 'prefix.nfstarb(gfsawaix)'
213504 bytes received in 2.4 seconds (87 Kbytes/s)
ftp> quit
221 Quit command received. Goodbye.
$
$ mkdir mvsnfs.client
$ cd mvsnfs.client
$ tar -xvf ../client.tarbin
tar: record size = 20 blocks
x ./makefile, 8234 bytes, 17 tape blocks
x ./gfsawaxd.c, 13474 bytes, 27 tape blocks
:
$
$ touch *.*
$ make ("make aix_rs " for RS/6000;"make solaris" for Solaris)
cc -c -I. gfsawsha.c
cc -c -I. gfsawaxd.c
cc -o showattr gfsawsho.o gfsawaxd.o
cc -c -I. gfsawlin.c
cc -c -I. gfsawclt.c
cc -c -I. gfsawmcl.c
cc -c -I. gfsawmou.c
cc -o mvslogin gfsawlin.o gfsawclt.o gfsawmcl.o gfsawmou.o
cc -c -I. gfsawlou.c
cc -o mvslogout gfsawlou.o gfsawclt.o gfsawmcl.o gfsawmou.o
$ ./mvslogin mvshost1 smith (MVS NFS must be operational
Password Required on host side.)
GFS A973A Enter MVS password: password
GFS A955I smith logged in ok.
$

```

Figure 6. Retrieving the client enabling commands for AIX, Sun Solaris, and HP/UX, UNIX (Part 2 of 2)

After retrieving the files of client enabling source code for the AIX or UNIX environment, follow these steps as shown in Figure 6 on page 33 to create the executable commands:

1. To make sure the source files have the current date. Some platforms do not have correct time stamps and cause make files to fail; issue command:

```
touch *.*
```

2. To create the executable commands **mvslogin**, **mvslogout**, and **showattr**, issue one of these commands for the environment which IBM supports:
 - make aix_rs** - Creates executable files for AIX on the RS/6000.
 - make aix_rs3** - Creates executable files for AIX 4.2.0 or higher for NFS version 3 support.

make solaris - Creates executable files for Sun Solaris 2.3 (SunOS 5.3) or Sun Solaris 2.5.1 (SunOS 5.5.1) with the following setup environment:

```
setenv LD_LIBRARY_PATH /usr/ucblib
```

make - Creates Sun executable files.

Retrieving commands for DEC ULTRIX

Figure 7 shows how to retrieve the necessary files to install the client enabling commands on a DEC ULTRIX client, where **mvshost1** is the name of the MVS host and **smith** is an MVS user ID:

```
# mkdir mvsnfs
# cd mvsnfs
$ ftp mvshost1
Connected to mvshost1
220-FTPSERVE at MVSHOST1, 01:44:24 on 10/02/94
220 Connection closes if idle for more than 5 minutes.
Name (mvshost1:w42dept): smith
<Press ENTER key>
331 Send password please.
Password: password
230 smith is logged on.
ftp> i
200 Type set to I
ftp> get 'prefix.nfstarb(gfsawaix)' client.tarbin
200 Port request OK.
125 Sending data set PREFIX.NFSTARB(GFSAWAIX)
250 Transfer complete
local: client.tarbin remote: 'prefix.nfstarb(gfsawaix)'
213504 bytes received in 2.4 seconds (87 Kbytes/s)
ftp> quit
221 Quit command received. Goodbye.
$
$ mkdir nfs.commands
$ cd nfs.commands
$ tar -xvf ../gfsawaix.tarbin .
x ./gfsawaxd.c, 13474 bytes, 27 blocks
x ./gfsawclt.c, 10735 bytes, 21 blocks
x ./gfsawlin.c, 18658 bytes, 37 blocks
x ./gfsawlou.c, 11759 bytes, 23 blocks
x ./gfsawmcl.c, 11249 bytes, 22 blocks
x ./gfsawmou.c, 27800 bytes, 55 blocks
x ./gfsawsha.c, 83061 bytes, 163 blocks
x ./gfsawxdr.c, 8696 bytes, 17 blocks
```

Figure 7. Retrieving the client enabling commands for DEC ULTRIX (Part 1 of 2)

```

x ./gfsawmnt.h, 7080 bytes, 14 blocks
x ./gfsawsho.h, 5286 bytes, 11 blocks
x ./makefile, 9929 bytes, 20 blocks
$ touch *.*
$ make ultrix
cc -c -I. gfsawaxd.c
cc -c -DULTRIX gfsawsha.c
cc -o showattr gfsawsha.o gfsawaxd.o
cc -c -I. gfsawlin.c
cc -c -I. gfsawclt.c
cc -c -I. gfsawmcl.c
cc -c -I. gfsawmou.c
cc -o mvslogin gfsawlin.o gfsawclt.o gfsawmcl.o gfsawmou.o
cc -c -I. gfsawlou.c
cc -o mvslogout gfsawlou.o gfsawclt.o gfsawmcl.o gfsawmou.o
$ ./mvslogin mvshost1 smith (MVS NFS must be
Password Required operational on host side.)
GFS A973A Enter MVS password: password
GFS A955I smith logged in ok.
$

```

Figure 7. Retrieving the client enabling commands for DEC ULTRIX (Part 2 of 2)

After retrieving the files of client enabling source code for the ULTRIX environment, follow these steps as shown in Figure 7 on page 35 to create the executable commands:

1. Enter the **touch *.*** command to make sure the source files have the current date because some platforms do not have correct time stamps and cause make files to fail.
2. Enter the following command to create the executable commands, **mvslogin**, **mvslogout**, and **showattr** for the ULTRIX environment (to create ULTRIX version executable files):

```
make ultrix
```

Retrieving commands for Sun PC-NFS

Figure 8 on page 37 shows how to retrieve the necessary files to install the client commands on a workstation with Sun PC-NFS Version 5.0, where **mvshost1** is the name of the MVS host and **smith** is an MVS user ID:

```

C> md client      (to create a directory to contain
                  client code)

C> cd client
C:\CLIENT> ftp mvshost1
Connected to mvshost1
220-FTPSERVE at MVSHOST1, 01:44:24 on 6/02/94
220 Connection closes if idle for more than 5 minutes.
Name (mvshost1:nobody): smith
<Press ENTER key>
331 Send password please.
Password: password
230 smith is logged on.
ftp> bin
200 Representation type is IMAGE.
ftp> get 'prefix.nfstarb(gfsawdin)' mvslogin.exe
200 Port request OK.
125 Sending data set PREFIX.NFSTARB(GFSAWDIN)
250 Transfer completed successfully.
local: mvslogin.exe remote: 'prefix.nfstarb(gfsawdin)'
70701 bytes received in 10.082000 seconds (6.85 Kbytes/s)
ftp> get 'prefix.nfstarb(gfsawdou)' mvslogout.exe
200 Port request OK.
125 Sending data set PREFIX.NFSTARB(GFSAWDOU)
250 Transfer completed successfully.
local: mvslogout.exe remote: 'prefix.nfstarb(gfsawdou)'
61821 bytes received in 5.011000 seconds (12.05 Kbytes/s)
ftp> get 'prefix.nfstarb(gfsawdsh)' showattr.exe
200 Port request OK.
125 Sending data set PREFIX.NFSTARB(GFSAWDSSH)
250 Transfer completed successfully.
local: showattr.exe remote: 'prefix.nfstarb(gfsawdsh)'
96027 bytes received in 5.032100 seconds (18.29 Kbytes/s)
ftp> quit
221 Quit command received. Goodbye.
C:\CLIENT>
C> (Set up the path to the "client" directory, or copy *.exe
    files to the path to enable the users to use the following client
    executable files: mvslogin.exe, mvslogout.exe, and showattr.exe)
C> mvslogin mvshost1 smith (MVS NFS must be operational
    Password required                                     on host side.)
GFS A973A Enter MVS password: password
GFS A955I smith logged in ok.
C>

```

Figure 8. Retrieving the client enabling commands for Sun PC-NFS version 5.0

Porting the mvslogin, mvslogout, and showattr commands

The z/OS NFS server supports any client machine that has an NFS client software package implemented according to the Sun NFS protocol. The z/OS NFS server has been tested with the following client platforms:

- SunOS 3 and 4
- AIX (on the RS/6000)
- Sun PC-NFS Version 5.0
- HP/UX Version 9.01
- Sun Solaris Version 2.3 (SunOS Version 5.3)
- Sun Solaris Version 2.5.1 (SunOS Version 5.5.1)
- DEC ULTRIX version 4.4
- OS/390 NFS
- z/OS NFS

The NFS supports the authentication procedures of PCNFSD Version 1 and Version 2 protocols, so there is no need to port the **mvslogin** and **mvslogout** commands for PC client platforms with PCNFSD client support if the PC client platforms keep the UID and GID to each mount point.

To port the **mvslogin**, **mvslogout**, and **showattr** commands successfully, you should understand the following:

- C language - The source code for these commands is written in C.
- System calls for your client machine's operating system - For example, the FAT file system under DOS only allows up to eight characters for file names, and up to three characters for file name extensions. AIX and UNIX do not have this restriction. Therefore, while **mvslogout** is a valid file name in an AIX or UNIX environment, it is too long to be a valid file name in a FAT file system under DOS.

As another example, the way that you get mount information varies for different platforms. The **mount** command is in the following (or similar) format:

```
mount <server>:<remote file system> <local mount point>
```

The minimum information for porting the client enabling commands is:

1. Server name
2. Remote file system (high-level qualifier)
3. Local Mount point
4. UID and GID

The system calls to get the information for porting the client enabling commands are platform-dependent. If you cannot find the information in the following types of documents for the platform, you must call the support telephone number for the platform and ask to speak with their NFS development department:

- Operating system development toolkit
- TCP/IP development toolkit
- NFS development toolkit
- The source code for **mvslogin**, **mvslogout**, and **showattr**.
For example, **mvslogin** tells the server the MVS user ID and its associated client UID number. This client UID number is expected to be passed to the server for all further client requests to the NFS. If the client user does not specify the MVS user ID and password on the **mvslogin** command, the MVS login ID is taken from the login ID on the workstation with no password assumed. If authentication for this default login ID from the workstation fails, then **mvslogin** prompts the user to enter the MVS login password.

Table 8 shows the common source files for the **mvslogin**, **mvslogout** (or **mvslogut**), and **showattr** commands on all platforms:

Seven.c files		
	gfsawaxd.c	XDR encode and decode routines for attributes service
	gfsawclt.c	NFS protocol calls for mvslogin and mvslogout
	gfsawlin.c	Main program to generate mvslogin command
	gfsawlou.c	Main program to generate mvslogout or mvslogut command
	gfsawmcl.c	Create the client handle and initialize it
	gfsawmou.c	XDR protocol definitions for mvslogin and mvslogout

Table 8. *Mvslogin, mvslogout, (or mvslogut), and showattr commands - all platforms (continued)*

	gfsawsha.c	Main program to generate showattr command mvslogut
Two .h files	gfsawmnt.h gfsawsho.h	Protocol definitions for mvslogin and mvslogut Attribute definition and procedures
One makefile		

Porting considerations

Procedures for porting vary for different C compilers and operating systems. Differences can occur during compiling, linking, and run time.

Compiling

The following items might vary for your client machine's operating system:

- Different set of compilation flags

There are different sets of compilation flags based on compilers or operating systems. For example:

 - AIX (on RS/6000) has the unique flags `_BSD`, `_SUN`, and `BSD_INCLUDES`.
 - DOS compilers have different compiler models, which require the corresponding compiler flag (for example, `-AL` and `-AS`).
- Include files in different directories

Because the include files can be installed differently based on the operating systems and their toolkits, the include files could be in different directories.
- Include file has a different name

Include files for the same or similar functions could have the same or similar file names. For example, DOS uses the file name "string.h", and the other platforms use "strings.h".
- System variables in different include files

The system variables are usually in different include files, based on the operating systems. For example, to access the mount table,

 - Sun PC-NFS Version 5.0 uses `dos.h`
 - Some other AIX and UNIX clients use `mntent.h`
- System variables have different names

The variables related to operating systems could have different variable names. For example; some AIX and UNIX clients use `getuid` to get the real UID.
- System variables have different structure

The structures related to operating systems could be different. For example, DOS FAT file systems have file name length restrictions which cause them to have a different directory structure from AIX or UNIX.
- System variables not supported

Some system variables are supported by one operating system but not another. For example, Sun PC-NFS Version 5.0 does not support `setuid`. AIX or UNIX supports both `getuid` and `getpwuid`.
- Sequence of include files

Some include files are based on the precedence of another include file. The prerequisite include file has to come before the other include files. For example, some Programming Libraries offer `types.h` which is based on C compiler `sys/types.h`. Therefore, `#include <sys/types.h>` should be before `#include <types.h>`.

- Mount information varies depending on the client operating system
The information about mount points provided by vendors of the client operating systems and client TCP/IP products varies and might not always be complete. To find the mount information:
 1. Search through the documentation (for example, the TCP/IP development toolkit and the installation and administration guides).
 2. If you cannot find the mount information in the documentation, contact the vendor that offers the TCP/IP development toolkit.

Linking

After linking the programs together, check for attention messages and error messages. The following items might vary for your client machine's operating system:

- Different set of linker/loader flags
Some programs require a different set of linker/loader flags. For example, the **mvslogin**, **mvslogout**, and **showattr** commands cannot use the "/NOI" option with the Sun PC-NFS Version 5.0 programmer's toolkit, version 1.0.
- Library files in a different directory
The library files required to complete the linkage could be in a different directory. For example, "OS2.LIB could be under the os2 directory rather than in the OS/2 toolkit directory.
- Library files have different names
Depending on how the client machines' operating systems are installed, the library files might have different names. For example, a C compiler could use "slibcer" for DOS/real mode, and use "slibcep" for OS/2 protected mode with these names set up during installation of the C compiler.
- Different libraries required
The system variables could be in different libraries for the different operating systems. For example, SunOS requires the rpcsvc library for some RPC calls, but some AIX clients do not require special libraries.
- Compiler is not compatible with the system toolkit
Some operating systems support multiple versions of C compilers. Some C compilers, however, might not match the various system toolkits. For example, the **mvslogin**, **mvslogut**, and **showattr** commands running under Sun PC-NFS Version 5.0 (on DOS) can be compiled with Microsoft® C 6.0A, but not with other compilers.
- Different library model required
The library models have to match with the compilation time. For example, for DOS because the compiling time uses the small model (the size of code and data is under 64K), the link time has to use the small model, too.
- System variables not supported
Some system variables are supported during compile time but not supported by the link time. For example, some XDR routines (xdr_fhandle, xdr_fhstatus, and xdr_path) are supported by AIX and UNIX, but for AIX for the RS/6000 they compile successfully but are not supported during link time.

Run time

After the compilation and linkage are successful, run the command to see if the result is as expected. If not, figure out the difference of the result or failure. The difference or failure can be in the following areas:

- Definition of the standard C variables is different

The definition of the standard C variables could be different for the different operating systems or compilers. Some special handling might be required. For example:

- *int* is 2 bytes long for DOS but it is 4 bytes long for the MVS NFS, AIX, and Sun.
- *tab* has a different value, causing the spacing of the output to be different.
- Definition of function calls is slightly different
Although a given function is supported, it might work slightly differently. For example, the “mount table” has a different format in AIX (for RS/6000) from SunOS.
- Library functions might have a defect
Some functions in the C libraries do not function as the documentation describes. You might report the problems or write your own functions to replace them.

Chapter 3. Attributes for the z/OS NFS server

This chapter contains information about the attributes that are used to manipulate files in the z/OS NFS server.

Table 9 contains directive information about this chapter's contents:

Table 9. Attributes - z/OS NFS server

Section	Modification	Description	Page
Data set creation attributes	Data set creation attributes can be modified by the client	Data set creation attributes provide information about an MVS file to the z/OS NFS server, such as the type of file, or how the file is allocated (for example, blocks, cylinders, or tracks)	44
Processing attributes	Processing attributes can be modified by the client	Processing attributes provide information to the z/OS NFS server about how to handle the file, such as how long the files remain open, or whether the files are processed in text or binary format	46
Site attributes	Site attributes can only be modified by the system administrator	Site attributes control the z/OS NFS server resources	54

Attributes used for z/OS UNIX System Services file access

These attributes are specific to z/OS UNIX file access:

- `hfs(prefix)`
- `sync` and `async`
- `extlink`

These attributes are relevant to accessing z/OS UNIX files as well as conventional MVS data sets:

- `restimeout` - Resource timeout
- `logout` - User log time out
- `security` - Security checking
- `text` - ASCII to EBCDIC data conversion and vice versa
- `binary` - No ASCII and EBCDIC
- `xlat` - Customized translation table

Using multipliers

Instead of entering the entire numeric values for the attributes, you can use the multipliers K (1024), M (1024 × 1024), or G (1024 × 1024 × 1024). For example, entering 10M is the same as entering 10,485,760.

Specifying attributes multiple times

Specifying an attribute several times on a line does not cause an error. The line is read from left to right, and the last of any duplicate attribute is used. For example:

```
$ vi "file,recfm(vb),recfm(fb)"
```

This results in a file created with a fixed-blocked format.

Data set creation attributes

The data set creation attributes are used to define the structure of MVS data sets when creating a file. These attributes correspond to the data control block (DCB) or the job control language (JCL) parameters used to define an MVS data set when it is created. See *z/OS MVS JCL Reference* for more information about data set creation attributes.

The data set creation attributes are described in Table 10. The data set creation attributes do not apply for HFS data sets. Defaults are underlined **in this format**.

You can override these attributes by using the **mount** command or file creation command. For PDS and PDSE, members have the same attributes as the data set attributes, so the file creation attributes specified for members are ignored.

Table 10. Data set creation attributes

Data Set Creation Attribute	Description
blks	Specifies that disk space (see the space attribute in this table) is allocated by blocks, except for VSAM data sets.
cyls	Specifies that disk space is allocated by cylinders.
recl	Specifies that disk space is allocated by records for VSAM data sets. blks and recl are identical for VSAM data sets.
trks	Specifies that disk space is allocated by tracks.
blksize(0 <i>quan</i>)	Specifies the maximum length, in bytes, of a physical block on disk. <i>quan</i> is a number from <u>0</u> (the default) to 32,760. If blksize(0) is specified, the system determines an optimal block size to use.
dataclas(<i>class_name</i>)	Specifies the data class associated with the file creation. The <i>class_name</i> must be defined to DFSMS/MVS® before it can be used by the client. The system-managed storage automatic class selection routine must also assign a storage class to the file being created. For more information about data classes, see <i>z/OS DFSMSdfp Storage Administration Reference</i> .
dir(27 <i>quan</i>)	Specifies the number of 256-byte records needed in the directory of a PDS. Use it with the mkdir command when you are creating a PDS. <i>quan</i> is a number from 1 to 16,777,215 (the default is <u>27</u>). The maximum number of PDS members is 14,562.
dsntype(<i>library</i> pds)	Specifies whether a PDSE or a PDS is to be created when the mkdir client command is used. library is for PDSE. pds is for PDS. You cannot create a PDS (or PDSE) within another PDS (or PDSE). If you need help deciding whether to create a PDS or a PDSE, see <i>z/OS DFSMS: Using Data Sets</i> .
dsorg(<i>org</i>)	Specifies the organization of a data set. <i>org</i> can be a physical sequential (ps) data set, direct access (DA) data set, VSAM KSDS (indexed), VSAM RRDS (numbered) or VSAM ESDS (nonindexed). This attribute is ignored for directory-oriented client commands. If you are using VSAM data sets in binary mode with an AIX client, then nonindexed is recommended.

Table 10. Data set creation attributes (continued)

Data Set Creation Attribute	Description
keys (<i>len, off</i>)	Specifies the length and offset of the keys for VSAM KSDS data sets. Keys can only be specified when using dsorg(indexed) . <i>len</i> and <i>off</i> are specified in bytes. <i>len</i> is between 1 and 255 (the default is 64). <i>off</i> is between 0 and 32,760 (the default is 0). When you create a VSAM KSDS data set, the records you are loading into it must be keyed-sequenced or the write fails. Each write of the data set is treated like a first load, and requires that the records being loaded are in ascending key sequence.
lrecl (8196 <i>quan</i>)	Specifies: <ol style="list-style-type: none"> 1. The length, in bytes, for fixed-length records. 2. The maximum length, in bytes, for variable-length records. If the blksize attribute is specified, the value must be at least 4 bytes less than the blksize quantity.
	<i>quan</i> is a number from 1 to 32,760 (the default is 8196).
mgmtclas (<i>mgmt_class_name</i>)	Specifies the management class associated with the file creation. The <i>mgmt_class_name</i> must be defined to DFSMS/MVS before it can be used by the client. The system-managed storage automatic class selection (ACS) routine must also assign a storage class to the file being created. For more information on management classes, see <i>z/OS DFSMSdfp Storage Administration Reference</i> .
recfm (<i>cccc</i>)	Specifies the format and characteristics of the records in the data set. <i>cccc</i> can be 1 to 4 characters, in one of the following combinations:
	f fb fs fbs
	u
	v vb vs vbs
	Valid record format characters:
	<i>b</i> Blocked
	<i>f</i> Fixed-length records
	<i>s</i> Spanned for variable records, standard format for fixed records
	<i>u</i> Undefined-length records
	<i>v</i> Variable-length records
	In recfm , codes v , f and u are mutually exclusive. The s code is not allowed for a PDS or PDSE.
recordsize (<i>avg,max</i>)	The average and maximum record size for VSAM data sets. <i>avg</i> and <i>max</i> are specified in bytes. They can each range from 1 to 32,760 (the defaults are 512 and 4096 , respectively). These values must be equal for VSAM RRDS.
rlse	Specifies that unused space should be released from the data set the first time a new data set is closed. For slow clients with long pauses between writes, the rlse attribute causes space to be released from the primary extent prematurely. Further writes cause secondary space to be allocated.
norlse	Specifies that unused space should not be released from the data set.
shareoptions (<i>xreg,xsys</i>)	Specifies the cross-region and cross-system share options for a VSAM data set. <i>xreg</i> is a number from 1 to 4; <i>xsys</i> is either 3 or 4. The defaults are 1 and 3 , respectively.

Table 10. Data set creation attributes (continued)

Data Set Creation Attribute	Description
	This applies to VSAM data sets only. For spanned records of non-VSAM data sets, see the entry for recfm in this table.
spanned	Specifies that VSAM KSDS or ESDS data sets can contain records that span control intervals (spanned records).
nonspanned	Specifies that data sets do not have spanned records.
space (<i>prim</i> [, <i>aux</i>])	Specifies the amount of primary and auxiliary space allocated for a new data set on a direct access volume. <i>prim</i> is the number (from 0 to 16,777,215) of primary tracks, cylinders, or data blocks in the data set. <i>aux</i> (optional) is the number (from 0 to 16,777,215) of additional tracks, cylinders, or blocks allocated if more space is needed. If this attribute is not specified, the default is used. The defaults are <u>100</u> and <u>10</u> , respectively.
storclas (<i>class_name</i>)	Specifies the storage class associated with the file creation. The <i>class_name</i> must be defined to the DFSMS/MVS before it can be used by the client. For more information about storage classes, see z/OS DFSMSdfp <i>Storage Administration Reference</i> .
unit (<i>unit_name</i>)	Specifies the unit on which to create a data set. <i>unit_name</i> is a generic or symbolic name of a group of DASD devices. The <i>unit_name</i> must be specified as 3390 for extended format data sets. Note: You cannot create or access tape data sets on an MVS host using the z/OS NFS server. You cannot create extended format data sets with the z/OS NFS server, except using ACS routines.
vol (<i>volser</i>)	Specifies the name of the DASD volume to be used to store the created data set. vol is the keyword and <i>volser</i> represents the volume name. If a data set is to be system-managed, as is determined by the DFSMS/MVS Automatic Class Selection (ACS) routines, you can omit this attribute.

Processing attributes

Processing attributes are used to control how files are accessed by the client. The processing attributes are described in Table 11. Defaults are underlined **in this format**. You can override the default processing attributes.

Table 11. Processing attributes

Processing Attribute	Description
attrtimeout (<i>n</i>)	The time (in seconds) that the data set remains allocated after a lookup or getattr server operation. The default is <u>120</u> . <i>n</i> can range from 1 to 32,767 (9 hours, 6 minutes, and 7 seconds). Note: lookup is an NFS protocol that searches for a file in the current directory. If it finds the file, lookup returns information on the file's attributes and a file handle pointing to the file. The attrtimeout value is normally greater than the readtimeout or writetimeout values.
noattrtimeout	The data set is not deallocated after a lookup or getattr operation. For more information, see "Timeout attributes" on page 50, following this table.

Table 11. Processing attributes (continued)

Processing Attribute	Description
binary	Indicates that the data set is processed between the client and server using binary format and no data conversion occurs between ASCII and EBCDIC formats.
text	<p>Converts the contents in the data set between EBCDIC and ASCII formats. Use this format to share text data between clients and MVS applications.</p> <p>In text mode, the following attributes apply only to conventional MVS data sets:</p> <ul style="list-style-type: none"> • blankstrip and noblankstrip. See the entry for blankstrip in this table. • End-of-line specifiers (lf, cr, lfcr, crlf, or noeol) are used to indicate the MVS logical record boundary. See the entry for lf in this table. See the xlat attribute in this table for customized EBCDIC-ASCII tables.
blankstrip	<p>With text mode, strips trailing blanks at the end of each record of a fixed-length text file when the file is read. Pads the end of each file or record with blanks when a text file is written.</p>
noblankstrip	<p>Does not strip trailing blanks at the end of fixed-length records when a fixed-length text file is read. Does not pad records when writing a text file. The file must be of the correct size or an I/O error is reported to the client.</p> <p>For information on the text attribute, see the entry for binary in this table.</p> <p>This attribute does not apply to HFS data sets.</p>
	<p>With text mode, use one of the following end-of-line specifiers:</p>
lf	Line Feed is the end-of-line terminator (standard AIX or UNIX).
cr	Carriage Return is the end-of-line terminator.
lfcr	Line Feed followed by Carriage Return is the end-of-line terminator.
crlf	Carriage Return followed by Line Feed is the end-of-line terminator (standard DOS).
noeol	No end-of-line terminator.
	<p>For information on the text attribute, see the entry for binary in this table.</p> <p>This attribute does not apply to HFS data sets.</p>
cln_ccsid(n)	<p>Specifies the Coded Character Set Identifier (CCSID) for the remote mounted file system (NFS client) when text is being translated.</p> <p>The default is 819 (ISO 8859-1 ASCII).</p> <p>This attribute applies only to hierarchical file system (HFS) files.</p>
executebiton	<p>Turns on the execute bits in user, group, and other (as reported with the ls (list) AIX or UNIX command) for a mount point's files. Use when storing executable or shell scripts on the MVS system. This option can only be overridden on a mount point basis — not at a command level. This attribute does not apply to HFS files and can only be used with the mount command.</p>
executebitoff	<p>Turns off the execute bits in user, group, and other for the mount point's files. This value is normally used in the site file.</p>

Table 11. Processing attributes (continued)

Processing Attribute	Description
extlink	Specifies the use of the external link command to create, process, and delete a symbolic link to an MVS data set. Is used with the ln -s command to create a symbolic link to an MVS data set. Is used with the ls -l command to display the attributes and contents of the symbolic link. Is used with the rm command to delete the symbolic link. This extlink attribute only applies to HFS file objects.
fastfilesize	For Direct Access data sets, PDSs, and non-system-managed data sets, this specifies to get the file size from SPF statistics, if they exist. For more information, see "Using fastfilesize to avoid read-for-xize" on page 182. This attribute also applies to PDSEs, but does not apply to HFS files.
nofastfilesize	For Direct Access data sets, PDSs, and non-system-managed data sets, this specifies to read the entire file or member to get the file size. Using this attribute might cause a noticeable delay when first accessing very large data sets. For more information, see "Using fastfilesize to avoid read-for-xize" on page 182. This attribute also currently applies to PDSEs.
fileextmap	Turns on file extension mapping. This option can be specified at the file command level for the client platforms that support passing of attributes. The default is nofileextmap .
nofileextmap	Turns file extension mapping off.
mapleaddot	Turns on mapping of a single leading "." from a client file name to a legal leading "\$" on MVS. This option would normally be enabled for access by AIX and UNIX clients. This attribute does not apply to HFS data sets.
nomapleaddot	Turns off mapping of a single leading "." from a client to a leading "\$" on MVS.
maplower	Turns on mapping of lower case file names to upper case when accessing files on MVS, and back when sending to the network. This option would normally be enabled for access by AIX or UNIX clients. This option only affects file names (high-level qualifiers and user catalog aliases). This attribute does not apply to HFS data sets.
nomaplower	Turns off mapping of lower case file names to upper case and back when using files on MVS. Exports data set entries are not translated to upper case when this option is specified in the default attributes. All mount requests are case sensitive.
mapped	This attribute should be specified at the mount or site level when a mixed set of data types is to be processed under a single mount point. The determination of whether the data is to be processed as text or binary depends on the rules that are established in the specified sidefile. If a file extension is not mapped to text or as binary using the sidefiles, then the data will be processed according to what has been specified as binary or text at the mount level, and finally, at the site level. If binary or text is specified at the file command level, the specification overrides the mapped specification.
rdrverf	Do cookie verifier checking for NFS version 3 <i>readdir</i> and <i>readdirplus</i> requests.
nordrverf	Do not do cookie verifier checking for NFS version 3 <i>readdir</i> and <i>readdirplus</i> requests.

Table 11. Processing attributes (continued)

Processing Attribute	Description
readtimeout(<i>n</i>)	<p>The amount of time in seconds before a data set is released after a read operation. <i>n</i> can range from 1 to 32,767 (9 hours, 6 minutes, and 7 seconds). The default is 90. The server closes the file when the file times out.</p> <p>This attribute does not apply to HFS.</p>
noreadtimeout	<p>The data set is not deallocated after a read operation.</p> <p>For more information, see “Timeout attributes” on page 50, following this table.</p>
	<p>The z/OS NFS server uses DFSMSHsm™ to recall or delete migrated files. The action that the server takes against the migrated files depends on which of the retrieve or noretrieve attributes is active. These attributes do not apply to HFS data sets.</p>
retrieve	<p>When the retrieve attribute is active, the server will recall the migrated file if necessary, upon an NFS_LOOKUP request for the file, depending on the file's status. The server may be able to obtain the migrated files attributes without recall (see “Retrieve attributes” on page 51 for additional information); if not the recall operation is started by the server. The server waits for the recall operation to complete if the file resides on DASD; if the file does not reside on DASD, the server does not wait for the recall operation to complete and returns a “device not available” message. You can try accessing the file later when the recall has completed.</p>
retrieve(wait)	<p>When the retrieve(wait) attribute is active, the server waits for the recall to finish.</p>
retrieve(nowait)	<p>When the retrieve(nowait) attribute is active, the server does not wait for the recall to finish, and immediately returns a “device not available” message. You can try accessing the file later when the recall has completed.</p>
noretrieve	<p>When the noretrieve attribute is active, the server does not recall the file, and an return “device not available” upon an NFS_LOOKUP, an NFS_READ, or an NFS_CREATE request for a file.</p> <p>For more information, see “Retrieve attributes” on page 51, following this table.</p>
setownerroot	<p>Sets the user ID in a file's attributes to root for a superuser. This attribute can only be used with mount command and does not apply to HFS.</p>
setownernobody	<p>Sets the user ID in a file's attributes to nobody, for a superuser.</p>
sidefile(<i>dsname</i>)	<p>Specifies the name of the data set that contains the rules for file extension mapping purposes. If a sidefile name is specified in the attributes data set, then it is the default sidefile for the NFS server. A user can also specify an additional sidefile name during a mount operation to be used along with the default. The mapping rules will first be searched in the sidefile specified during the mount command and then the default sidefile is searched. To allow file extension mapping, a sidefile name must be specified either as a default or in the mount command. <i>dsname</i> is a fully-qualified MVS data set name without quotation marks. <i>sidefile</i> is only specified at the mount level. See GFSAPMAP in NFSSAMP for sample sidefile and syntax rules. This attribute does not apply to HFS.</p>

Table 11. Processing attributes (continued)

Processing Attribute	Description
srv_ccsid(n)	<p>Specifies the Coded Character Set Identifier (CCSID) for the local mounted file system (NFS server) when a new file is being created.</p> <p>This attribute has no effect on the translation of existing files' data.</p> <p>The default (if specified) is 1047 (LATIN OPEN SYSTEM EBCDIC).</p> <p>If this attribute is not specified, new hierarchical file system (HFS) files will continue to be created as untagged.</p> <p>This attribute applies only to HFS files.</p>
sync	<p>Specifies that data transmitted with the write request should be committed to nonvolatile media (for example, DASD) by the server when received.</p>
async	<p>The user can alternatively specify the async processing attribute to get improved performance.</p> <p>The sync async attribute only applies to HFS file objects and the NFS version 2 protocol.</p>
writetimeout(n)	<p>Specifies the amount of time, in seconds, before a data set is released after a write operation. <i>n</i> can range from 1 to 32,767 (9 hours, 6 minutes, and 7 seconds). The default is 30. The server closes the file when the file times out. All cached buffers are forced to disk. Normally writetimeout values are kept short because WRITE operations result in exclusive locking. However, for slow client machines with long pauses between writes, you should increase the writetimeout value.</p> <p>This attribute does not apply to HFS.</p>
nowritetimeout	<p>Specifies that the data set is not deallocated after a write operation.</p> <p>For more information, see "Timeout attributes", following this table.</p>
xlat(member_name)	<p>This attribute can be used to override the installation default translation table during file processing. <i>member_name</i> is the member name of the PDS or PDSE that contains the customized translation table. The system administrator defines this member name in the attribute data set, and PDS or PDSE in the startup procedure. This attribute is ignored if specified on the command line. See Customizing the Translation Table in the z/OS NFS Customization and Operation Manual for more information.</p> <p>If a customized translation table is not specified in the attribute file or in the mount command, xlat() is displayed by the showattrclient enabling command.</p>

Timeout attributes

The values of the following attributes depend on the settings of the associated site attributes:

- The attributes **attrtimeout**, **readtimeout**, and **writetimeout** must be within the ranges specified by the **maxtimeout** and **mintimeout** site attributes.
- The attributes **noattrtimeout**, **noreadtimeout** and **nowritetimeout** are valid only when **nomaxtimeout** is specified in the site attributes.

There are three processing attributes which control when files are timed out: **attrtimeout**, **readtimeout**, and **writetimeout**. The server determines that of these timeouts are in effect based on the last file operation. Thus when an existing file is appended, the file cannot be accessed before it times out in the time specified for **writetimeout** and is released by the server, because write operations result in

exclusive locking. Similarly, if a file is read, it is not released before it times out in the time specified for **readtimeout** seconds.

The new **readdirtimeout** site attribute controls the internal **readdir** cache used by directory lookups of MVS conventional data sets to be timed out or discarded based on a customizable value. The default is 30 seconds.

Retrieve attributes

The server deletes the migrated file upon an NFS_REMOVE request for a file, regardless of whether the **retrieve** or the **noretrieve** attribute is active. Typically, an NFS_REMOVE request is preceded by an NFS_LOOKUP request. If the data set was migrated with DFSMS/MVS 1.2 or below, retrieve attribute causes a recall because NFS_LOOKUP processing needs to open the data set and read for size. If the data set was migrated under DFSMS/MVS 1.3 and DFSMSHsm 1.3, and is SMS managed, its attributes were saved on DASD; therefore it is not always necessary to recall the data set to read for size and the data set may be deleted without recall. If the **noretrieve** attribute is active, the NFS_LOOKUP can return a “device not available” message. If the client code decides to ignore the error and go with the NFS_REMOVE, the migrated file is then deleted.

The UNIX command **ls mvshost** does not issue requests for individual files under the mvshost directory. Migrated files under the mvshost directory are displayed, but are not recalled. However, the UNIX command **ls -l mvshost** issues NFS_LOOKUP requests for individual files under the mvshost directory.

Mapped keyword processing attribute

Table 12 contains *mapped* and existing keyword information:

Table 12. The mapped keyword and existing keywords

SFMAX	SIDFILE(NAME)	MAPPED	ACTION
=0	X	Don't care	Data processed using existing rules for binary/text
	I		Server won't come up
	M		MOUNT will fail

Table 12. The mapped keyword and existing keywords (continued)

SFMAX	SIDFILE(NAME)	MAPPED	ACTION
1–2000	I + M	SET	File extension used in the MOUNT-specified sidefile and then the site-specified sidefile. If an extension is not found, the existing rules for binary/text will be used.
		NOT SET	Data processed using existing rules for binary/text
	I	SET	File extension used in the site-specified sidefile. If an extension is not found, the existing rules for binary/text will be used.
		NOT SET	Data processed using existing rules for binary/text
	X	Don't care	Data processed using existing rules for binary/text
	M	SET	File extension used in the mount-specified sidefile. If an extension is not found, the existing rules for binary/text will be used
		NOT SET	Data processed using existing rules for binary/text

Legend:

I = sidefile specified in installation table
M = sidefile specified in mount command
X = no sidefile specified

Native ASCII processing attributes

The *cln_ccsid(n)* and *srv_ccsid(n)* attributes can be specified either as installation defaults or at mount time for more granularity between different mount points. Unless *srv_ccsid* is specified either as an installation default or at mount time, newly created files will not have any file tag set (i.e., file tag is all zeros). These two attributes affect translation only when text processing is involved and only when an existing file has a non-zero or a non-binary file tag.

Special attention must be paid to the different server attributes specified. See Table 13 on page 53 and Table 14 on page 53.

Considerations for Native ASCII environment support

For applications running on z/OS V1R2 (and higher), a Native ASCII environment is provided for hierarchical file system (HFS) file processing.

In this environment, applications can operate on files in either EBCDIC or ASCII format as well as other data formats defined with a Coded Character Set Identifier (CCSID) without translation, provided the data is already defined and stored in the data format wanted.

For the z/OS NFS server to operate properly on HFS files in this environment, consider the following important factors:

- Unicode Conversion Services must be installed and set up on the system to let the NFS server use it for text translation.

- Two new processing attributes, **cln_ccsid** and **srv_ccsid**, are available for the NFS server for translation purposes as well as for the creation of new files. The **srv_ccsid** attribute determines the CCSID of newly created HFS files. If **srv_ccsid** is not specified as an installation default or at mount time, then new files continue to be created as untagged, or with a tag of 0x0000 and the old translation method of using translation tables specified by the **xlat** keyword applies
- Processing (READ/WRITE) of tagged files depends on the different server options specified. See details in “Native ASCII processing attributes” on page 52.

Table 13 and Table 14 contain considerations for Native ASCII environment support.

NFS clients with non-390 based NFS servers: Table 13 contains NFS client options.

Table 13. z/OS NFS clients with non-390 based NFS servers

Client Options Specified	Mount Option	Read	Write
xlat(Y), cln_ccsid,srv_ccsid	No TAG specified	Same as today - NFS client does translation	Same as today - NFS client does translation
xlat(N)	TAG(TEXT,CCSID)	Logical file system does translation	Logical file system does translation
xlat(Y)	TAG(TEXT,CCSID)	Mount will fail	Mount will fail

Notes:

1. The logical file system will do translation when the **mount TAG** option is specified. It will do translation based on the process tag (calling application) and file tag (if the file tag is not zeros or untagged). Otherwise, the system will do translation based on process tag and mount TAG the CCSID information.
2. It is assumed that the user doing the **mount** knows the files being accessed from the remote non-390 file systems. So the CCSID needs to be set accordingly. Data written to the server will be stored in a specific CCSID format and to read it back correctly, the correct CCSID needs to be specified (for example, without it being translated with the wrong CCSID).

See *z/OS Network File System User's Guide* for more information about client mount options.

NFS servers with non-390 based NFS clients: Table 14 contains NFS server options. See “Native ASCII processing attributes” on page 52 for more information.

Table 14. NFS servers with non-390 based NFS clients

Server Options Specified	File Tag	Read	Write	Create
text <xlat(table_name)> ⁵	Untagged or Tag=0x0000	Translation using the current xlat tables	Translation using the current xlat tables	New file created with CCSID of 0x0000
text	Yes	Translation based on file tag and default cln_ccsid of 8859 ³	Translation based on file tag and default cln_ccsid of 8859 ³	New file created with CCSID of 0x0000
text xlat(table_name)	Yes	Fail operation	Fail operation	New file created with CCSID of 0x0000
text,cln_ccsid,<srv_ccsid><xlat(table_name)> ⁵	Untagged or Tag=0x0000	Translation using the current xlat tables	Translation using the current xlat tables	New file created with srv_ccsid tag ²
text,cln_ccsid,<srv_ccsid> ⁴	Yes	Translation based on file tag and cln_ccsid ³	Translation based on file tag and cln_ccsid ^{1,3}	New file created with srv_ccsid tag ²

Table 14. NFS servers with non-390 based NFS clients (continued)

Server Options Specified	File Tag	Read	Write	Create
binary	Untagged or Tag=0x0000	No translation	No translation	New file created with binary tag
binary	Yes	No translation	Fail operation unless file tag is binary or TXTFLAG=OFF in file tag	New file created with binary tag

Notes:

1. Writing to a file that has a tag that is different from the **srv_ccsid** (regardless of whether the file is empty or not) will result in the file tag overriding the specified **srv_ccsid** when text is specified.
2. If **srv_ccsid** is specified (as an installation default or at mount), then the file will be created with the **srv_ccsid** tag. Otherwise, an untagged file is created.
3. If TXTFLAG=OFF (that is, not pure text) with non-zero CCSID and text is specified, it is assumed that the user knows which part of the file is text and translation proceeds as requested by client. In order to access a file that has mixed data without the data being translated during read or write operation, the binary option must be used.
4. **xlat** (if specified) is ignored when the file being accessed is tagged.
5. **xlat** is optional. For untagged files, translation is done using default **xlat** tables or customized **xlat** tables (if specified).
6. There is no facility in the NFS server to change an existing file tag. This must be done outside of the NFS server.
7. Specifying the binary option overrides any **cln_ccsid** and **srv_ccsid** is specified.
8. All files created that are by the server when text and **srv_ccsid** are specified will also have the TXTFLAG set to ON.

z/OS NFS client with z/OS NFS server: Both the client and server operate as described in “NFS clients with non-390 based NFS servers” on page 53 and “NFS servers with non-390 based NFS clients” on page 53.

In order to avoid double translation, the mount to the server must specify the correct **cln_ccsid** (server option) and the client **tso mount** command should not have the **tag** option. The client mount option **xlat(N)** should be specified so that only the server will do translation (if needed) and return the data in the correct CCSID.

```
mount filesystem(NFS001) type(nfs) mountpoint('/u/nfsdir')
      parm('mvsnfs: "/u/hfs/u/user,text,cln_ccsid(2000)",xlat(N)')
      vi /u/nfsdir/file1

** Translation will be done based on file1's file tag and cln_ccsid of
   2000 by server only.
```

In all other cases, double translation may occur as the server will do translation based on its file tag and **cln_ccsid** settings and logical file system will do translation based on the process tag and the CCSID in the **mount TAG** option. Caution must be used as double translation may result in the data becoming garbage.

Site attributes

The site attributes are used to control z/OS NFS server resources. These attributes are described in Table 15 on page 55. Some initial settings are shown, but the system administrator might have changed these settings, so use the **showattr** command to show the actual settings being used. The site attributes cannot be modified by client users.

Table 15. Site attributes

Site Attribute	Description
bufhigh(<i>n</i>)	<p>Specifies the maximum size (in bytes) of allocated buffers before buffer reclamation (see the percentsteal attribute in this table) is initiated. <i>n</i> is an integer from 1 MB to 2047 MB (the default is 32 MB). If the combined total specified in the bufhigh and logicalcache attributes is greater than the available storage in the extended private area (implied by the REGION parameter in your procedure) at startup, the server shuts down immediately. A higher number means more caching and potentially better read performance.</p>
cachewindow(<i>n</i>)	<p>Specifies the window size used in logical I/O to buffer client block writes received out of order. <i>n</i> is a number from 1 to 256 (the default is 112). This attribute does not apply to HFS data sets. The suggested value is some small multiple of the number of BIODs running on a client. The general rule in setting the <i>n</i> value of cachewindow(<i>n</i>) is $n = ((\text{num of BIOD} + 1) * (\text{client_max_IO_buffer_size}/\text{transfer_size}))$ where,</p> <ul style="list-style-type: none"> • BIOD is the number of blocked I/O daemons set by the client workstation. This value is usually set to defaults at the installation of the operating system or by your system administrator. • client_max_IO_buffer_size is the amount of I/O data requested by the client (for example, client writes 8192 bytes of data to the remote file system). This value is determined by your application programs. • transfer_size is the actual size of data being sent across the network (for example, the 8192 bytes of data can be broken down to 16 smaller packets of 512 bytes (16x512=8192)). This value is determined dynamically by your client workstation.
checklist	<p>When specified, the server bypasses SAF checking (even when SAF or SAFEXP is specified) for the list of files and directories underneath mount points which either matches a mount point entry or is a child of a mount point entry in the CHKLIST DD data set. CHECKLIST is only valid if SAF checking is the security option for the particular data access; otherwise, it is ignored even if it is specified. See GFSAPCHK in NFSSAMP library for sample CHKLIST data set.</p>
<u>nochecklist</u>	<p>When specified, the server operates as before and ignores the information that is specified in the CHKLIST DD data set.</p>
fn_delimiter	<p>Specifies a character 'c' to be used as a delimiter between the file name and the attributes that follow it. This capability allows those sites that have UNIX data sets containing commas to copy and store their data on the NFS server. The following example specifies the default delimiter as a semicolon:</p>
<pre>fn_delimiter(;</pre>	
<p>So a user can process a file called 'comma,in-name' by entering:</p>	
<pre>vi "comma,in-name;text,lf"</pre>	
<p>A user can also include a default file name delimiter as a comma as follows:</p>	
<pre>fn_delimiter(,</pre>	
<u>fn_delimiter(,)</u>	<p>The default file name delimiter is a comma.</p>

Table 15. Site attributes (continued)

Site Attribute	Description
hfs(prefix)	<p>Specifies a new HFS file system <i>prefix</i> to be imbedded in the mount directory path name. The default value of the HFS file system <i>prefix</i> is /hfs. Mount requests received by the z/OS NFS server beginning with the HFS file system <i>prefix</i> value are identified as mount requests for z/OS UNIX. The HFS file system <i>prefix</i> value is not part of the path name.</p> <p>Notes:</p> <ol style="list-style-type: none"> 1. The HFS file system must be mounted locally by z/OS UNIX or the client mount fails. 2. The <i>prefix</i> value can only be 7 characters or less including the beginning "/"
nohfs	Disables Network File System z/OS UNIX operation.
hfsbtimeout	<p>Specifies how to control the timeout of the HFS vnode token used by the NFS server. The timeout value controls how long before vnode tokens saved in file blocks are released.</p> <p>The valid range is 1 to 32,767 seconds.</p> <p><i>n</i> can go as low as 1 second but to avoid the possibility of the client hanging (because of network delays), <i>n</i> is not recommended to be lower than 5 seconds. <i>n</i> may need to be increased if the network is slow and the accessed directory has a lot of entries.</p> <p>The default hfsbtimeout is 60 seconds.</p>
leadswitch	Tells the server to return '/' as the first character in each export entry.
noleadswitch	<p>Tells the server not to return '/' as the first character in each export entry.</p> <p>Note: The leadswitch attribute is ignored for HFS file objects.</p>
logicalcache(n)	<p>Specifies the maximum size (in bytes) of allocated buffers in the logical I/O processing for all the cache windows combined. <i>n</i> is an integer from 1 MB to 2047 MB (the default is 16 MB). The recommended value for this attribute is 50% of the bufhigh attribute. This attribute does not apply to HFS data sets.</p>
logout(n)	<p>Specifies the time limit for inactivity in seconds for a given user on a client (the default is 1800). When the limit is reached, the user is automatically logged out. The client user must enter the mvslogin command again to reestablish the client's MVS session. This value should normally be the same as the value defined for TSO logout at your site. <i>n</i> can range from 1 second to 20 megaseconds (approximately 243 days).</p>
maxrdforszleft(n)	<p>Defines the number of physical block buffers left after determining a file's size. This operation is done for later server read requests to the same file. The buffers left are subject to trimming during a "buffer steal" operation. <i>n</i> is an integer from 1 to 1024 (the default is 32).</p>
maxtimeout(n)	<p>Specifies the maximum timeout allowed. This attribute and the mintimeout attribute define the range of values that client users can specify for attrtimeout, readtimeout, and writetimeout. <i>n</i> is the number of seconds from 1 to 32,767 (9 hours, 6 minutes, and 7 seconds). This attribute does not affect the logout attribute.</p>
nomaxtimeout	Allows client users to specify noattrtimeout , noreadtimeout , and nowritetimeout .

Table 15. Site attributes (continued)

Site Attribute	Description
mintimeout(<i>n</i>)	Specifies the minimum timeout. This attribute and maxtimeout define the range of values that can be specified for attrtimeout , readtimeout , and writetimeout . <i>n</i> is the number of seconds from 1 to 32,767 (the default is <u>1</u>).
nfstasks(<i>n,m,o</i>)	<p>Specifies the number of server processes to initiate on startup. The valid value range for <i>n</i>, <i>m</i>, and <i>o</i> are as follows:</p> <p>If nfstasks(<i>n,m</i>) is specified, then the following is true: <i>n</i> is the number of subtasks that handle the asynchronous input/output (I/O) operations or short blocking operations (the maximum number of concurrent NFS server requests). <i>m</i> is the number of subtasks that handle the long blocking operations (the maximum number of concurrent NFS server recall and HFS requests). Increase this value if your server supports lots of active recall or HFS clients.</p> <p>The valid value range for both <i>n</i> and <i>m</i> is 1 to 24. The sum of <i>n</i> and <i>m</i> must be less than or equal to 25.</p> <p>If nfstasks(<i>n,m,o</i>) is specified, then the following is true: <i>n</i> is the number of subtasks that handle the asynchronous input/output (I/O) operations or short blocking operations (the maximum number of concurrent NFS server requests). <i>m</i> is the number of subtasks that handle HFS requests. Increase this value if your server supports lots of active HFS clients. <i>o</i> is the number of subtasks that handle the long blocking operations (the maximum number of concurrent NFS server recall requests). Increase this value if your server supports lots of active recall operations.</p> <ul style="list-style-type: none"> • The valid value range for <i>n</i> is 1 to 24. • The valid value range for <i>m</i> is 1 to 100. • The valid value range for <i>o</i> is 1 to 24. • The sum of <i>n</i> and <i>o</i> must be less than or equal to 25. <p>The default is nfstasks(8,16,8).</p>
pcnfsd	Tells the Network File System to start the PCNFSD server.
nopcnfsd	Tells the Network File System not to start the PCNFSD server.
percentsteal(<i>n</i>)	Specifies the percent of the buffers reclaimed for use when the bufhigh(<i>n</i>) limit has been reached. A higher value means a reclaim operation is performed less often, but the cached data is significantly trimmed on each reclaim. This can result in poor read performance because readahead buffers might be stolen. Lower values result in more frequent reclaim operations, but the cached data normal water mark is higher, meaning possibly better performance by reading out of cached data. <i>n</i> is an integer from 1 to 99 (the default is <u>20</u>).
public(legacy_path,hfs_path)	This attribute specifies the legacy path (MVS conventional data) and HFS path that is associated with the public file handle for WebNFS access. The first path, if specified, is the legacy path. The second path is the HFS path and must start with the HFS prefix specified in the HFS() keyword. If the first path is not there, a comma must precede the second path. If the <i>public</i> keyword is specified, then one of the paths must be specified. The <i>public</i> keyword must be specified after the HFS() keyword in the site attribute table. A lookup request with the <i>public</i> file handle determines which of the two paths it is referring to by the pathname that it comes in with. An absolute pathname will tell the server which of the paths it is referring to by matching one of the paths specified. A lookup request with a relative pathname will be taken to be an HFS request if HFS is active (hfs_path has been provided); otherwise, it is treated as a legacy request. The default is <u>no public path</u>

Table 15. Site attributes (continued)

Site Attribute	Description
readaheadmax(<i>n</i>)	<p>This attribute defines the number of bytes to be read to fill internal buffers during read processing to enhance satisfying read requests directly from cache. This reduces the amount of synchronous physical I/O performed for Network File System read requests for sequential read file access. It also reduces context switching overhead on Network File System read requests by allowing more read requests to be satisfied directly from the main task. <i>n</i> is an integer from 1 KB to 128 KB (normally 2 to 4 times the common block size used for file access, which is recommended at 8 KB for AIX file activity). The default is 16,384. Specifying zero (0) will deactivate readahead.</p>
readdirtimeout	<p>Specifies the amount of time, in seconds, before the internal <i>readdir</i> cache that is used for MVS conventional data set is timed out or discarded. Valid range is from 1 to 32,767 (9 hours, 6 minutes, and 7 seconds). <i>n</i> can go as low as 1 second, but to avoid the possibility of <i>client hanging</i> (due to network delays and staled cache), <i>n</i> is not recommended to be lower than 5 seconds. <i>n</i> may need to be increased if the network is slow and the accessed directory has a lot of entries.</p> <p>The default readdirtimeout is 30 seconds.</p>
restimeout(<i>n,m</i>)	<p>Specifies a retention period and a clock time for the removal of mount points and control blocks that have been inactive longer than the specified retention period.</p> <p><i>n</i> is the resource retention period for mounts and associated resources. If they have been inactive for more than <i>n</i> hours, they are removed. The valid range for <i>n</i> is 0 to 720 hours (30 days). The default is 48 hours. If <i>n</i> is set to 0, the z/OS NFS server does not remove any mount points or associated resources.</p> <p><i>m</i> is the time of day to do the cleanup for mounts and associated resources that have been inactive more than <i>n</i> hours. The time of day is specified as a 24 hour local time value. The valid range for <i>m</i> is 0 to 23. The default is 0 (that is, midnight).</p> <p>Because cleanup work slows down the server, set <i>m</i> so that cleanup work occurs when the server is lightly loaded. If a mount handle is removed by the cleanup activity, the user must do the umount and mount operations to access the mount point again. The resource cleanup is also done when the server is shutting down.</p>
security(mvs,hfs,public)	<p>Where MVS data sets (legacy data), HFS data, and public data (data that is accessed using the public file handle) are place holders for security options: NONE, SAF, SAFEXP, EXPORTS. The first parameter, mvs, specifies the security option for MVS conventional data. The second parameter, hfs, specifies the security option for HFS data. The third parameter, public, specifies the security option for data that is accessed with the public file handle. The mvs parameter is required and the hfs and public parameters are optional. When the optional parameters are not specified, they are assigned the same security option as the first parameter.</p> <p>Note: The security options are the same as before (NONE, SAF, SAFEXP, EXPORTS); however, now a different security option can be picked for each of the three different data accesses.</p> <p>The security options are as follows:</p> <p>none Neither SAF checking nor exports list checking. For HFS, checks UNIX permission bits; obtains the UID from the client RPC request.</p> <p>saf SAF checking. No exports checking. For HFS, checks UNIX permission bits; obtains the UID from the z/OS UNIX segment using mvslogin.</p> <p>exports Exports list checking. For HFS, checks UNIX permission bits for HFS data; obtains the UID from the client RPC request. No SAF checking.</p> <p>safexp SAF checking and exports list checking. For HFS, checks UNIX permission bits; obtains the UID from the z/OS UNIX segment using mvslogin.</p> <p>The default is security(SAFEXP,SAFEXP, SAFEXP)</p>

Table 15. Site attributes (continued)

Site Attribute	Description
sfmax(<i>n</i>)	Where <i>n</i> specifies the maximum size (in kilobytes) of allocated storage for all of the sidefiles. <i>n</i> is an integer from 0 to 2000. The default value is 0 and it signifies that no mapping is allowed on the NFS server. If sfmax=0, specifying sidefile keyword in the attributes data set will cause the server to shut down and the specifying the sidefile in any subsequent mount commands causes the mount to fail as mapping is not allowed on the NFS server. If the amount of storage specified cannot be obtained during server initialization then the server will shut down immediately.
smf(<u>none</u> user file userfile)	Specifies the SMF records to be recorded: none No SMF data collection user Collection of user session SMF data file Collection of file usage SMF data userfile Collection of file usage and user session SMF data

Chapter 4. Attributes for the z/OS NFS client

This chapter contains information about the attributes that are used by the z/OS NFS client.

Table 16 contains directive information about this chapter's contents:

Table 16. Attributes - z/OS NFS client

Section	Page
Client attributes	61
DataCaching attribute	63
Mount processing parameters and installation parameters	64

Client attributes

Client attributes are described in Table 17.

Table 17. Client attributes

Attribute	Description
acdirmax(n)	Specifies the maximum lifetime in seconds of cached file attributes. The default value is <u>60</u> .
acdirmin(n)	Specifies the minimum lifetime in seconds of cached file attributes. The default value is <u>30</u> .
acregmax(n)	Specifies the maximum lifetime in seconds of cached file attributes. The default value is <u>60</u> .
acregmin(n)	Specifies the minimum lifetime in seconds of cached file attributes. The default value is <u>3</u> .
AttrCaching(Y/N)	Specifies whether to process attributes and data caching. The default value is <u>Y</u> . If attribute caching is in effect, the z/OS NFS client maintains cache consistency with the copy of the file on the NFS server by performing the consistency check with the cached file attributes. When a file's data is read, it remains valid on the z/OS NFS client until the attribute cache is timed out or negated. If AttrCaching(N) is specified, it will automatically set DataCaching(N) .
Biod(n)	Specifies the number of asynchronous block input/output (I/O) daemons. The valid range is 1 to 32, the default value is <u>6</u> . The Biod daemon runs on all NFS client systems. When a user on a client wants to read or write to a file on a server, the Biod daemon sends this request to the server. The Biod daemon is activated during system startup and runs continuously. The number of daemons is based on the load the client can handle. Six to eight daemons can handle an average load. You must run at least one daemon for NFS to work.
Bufhigh(n)	Specifies the storage limit for the NFS client. The valid range is 1 MB to 128 MB. The default value is <u>32</u> MB.
cln_ccsid(x)	Specifies the Coded Character Set Identifier (CCSID) for the local mounted file system. The default is <u>1047</u> (LATIN OPEN SYSTEM EBCDIC).

Table 17. Client attributes (continued)

Attribute	Description
DataCaching(Y/N)	<p>Specifies whether to perform data caching. The default value is <u>Y</u>.</p> <p>The DataCaching attribute provides finer granularity in controlling whether file data should be cached by the z/OS NFS client. By caching the file data, all subsequent references to the cached data is done locally thus avoiding the network overhead. This has additional significance when obtaining data from Network File System server systems which do not use UNIX access permissions for security as there is a potential security exposure allowing unauthorized users to access file data.</p>
DelayWrite(n)	<p>Specifies the maximum number of disk blocks for delay write. The valid range is 0 to 32. The default value is <u>16</u>. The blocksize is 8192. This option is only valid when DataCaching=Y.</p>
Delim(binary nl cr lf crlf lfcrlf)	<p>Specifies the line delimiter for record access to remote file through basic sequential access method (BSAM), queued sequential access method (QSAM), and virtual storage access method (VSAM). The default value is binary.</p> <p><i>binary</i> specifies the data does not have record delimiters. The access method does not add a delimiter for each record on output and treats any delimiters on input as data.</p> <p>The following values specify text:</p> <p><i>cr</i> – Specifies that records are delimited by the EBCDIC carriage return character (x'0D').</p> <p><i>crlf</i> – Specifies that records are delimited by the EBCDIC carriage return character followed by the EBCDIC line feed character (x'0D25').</p> <p><i>crlf</i> – Specifies that records are delimited by the EBCDIC carriage return character followed by the EBCDIC new line character (x'0D15').</p> <p><i>lf</i> – Specifies that records are delimited by the EBCDIC line feed character (x'25').</p> <p><i>lfcrlf</i> – Specifies that records are delimited by the EBCDIC line feed character followed by the EBCDIC carriage return character (x'250D').</p> <p><i>nl</i> – Specifies that records are delimited by the EBCDIC new line character (x'15').</p>
DynamicSizeAdj(Y/N)	<p>Specifies whether to perform the packet size adjustment for remote procedure call (RPC). The default value is <u>Y</u>.</p>
proto(tcp udp)	<p>Specifies the transport protocol for the NFS client to communicate with the NFS server. If proto(tcp) is specified, the server does not support <i>tcp</i>; <i>udp</i> is used instead. By default, the NFS client selects the proto and vers with the following priorities:</p> <ol style="list-style-type: none"> 1. <i>proto(tcp)</i> and <i>vers(3)</i> 2. <i>proto(udp)</i> and <i>vers(3)</i> 3. <i>proto(tcp)</i> and <i>vers(2)</i> 4. <i>proto(udp)</i> and <i>vers(2)</i>
ReadAhead(n)	<p>Specifies the maximum number of disk blocks to read ahead. The block size is 8192 bytes. The valid range is 0 to 16. The default value is <u>1</u>.</p>
retrans(n)	<p>Sets the number of time to retransmit the NFS remote procedure calls (RPC). The valid range is 0 to 1000 and the default value is <u>3</u>. This option is only valid when <i>soft</i> is specified.</p>

Table 17. Client attributes (continued)

Attribute	Description
retry(<i>n</i>)	Specifies the number of time to retry the mount operation. The valid range is 0 to 20,000. The default value is 10000 . This option is only valid during mount processing.
resize(<i>n</i>)	Sets the read buffer size to <i>n</i> bytes. The valid range is 1 to 8192, and the default value is 8192 .
soft hard	Returns an error if the NFS server does not respond or continue to retry the NFS remote procedure call (RPC) until the NFS server responds. If you specify <i>hard</i> , then continue to retry NFS remote procedure call (RPC) until the NFS server responds. If <i>soft</i> is specified, an error is returned if the NFS server does not respond. Specify the maximum number of retries with the retrans option. The default value is hard . This option is valid for all NFS RPCs under the mount point.
srv_ccsid(<i>x</i>)	Specifies the Coded Character Set Identifier (CCSID) for the remote mounted file system. The default is 819 (ISO 8859–1 ASCII).
timeo(<i>n</i>)	Sets the remote procedure call (RPC) to <i>n</i> tenths of a second. The default value is 7 .
vers(2 3)	Specifies the NFS protocol version that the client will use to communicate with the NFS server. If no version is specified, the z/OS NFS client will communicate with the NFS server at the highest protocol level that is supported by the server.
wsize(<i>n</i>)	Sets the write buffer size to <i>n</i> bytes. The default value is 8192 .
xlat(Y N)	If Y is specified, the data in all the files are text and the NFS client will perform data conversion according to the <i>cln_ccsid</i> and <i>srv_ccsid</i> parameters. The default value is N and should be used for binary data.

DataCaching attribute

Security checking is done on the Network File Systemserver to determine whether the requesting client user is authorized to access the data. On UNIX systems, this is done by validating the client's user ID and group ID against the file's permission codes. If the authorization checking is successful, the file data is returned to the z/OS NFS client system. Further authorization checking for subsequent access to the cached data or for other client users is done on the z/OS NFS client system.

For conventional MVS data set access through the z/OS NFS server, the user is required to present their MVS credentials which are checked by the MVS security system, such as RACF, before file data is returned. Since the MVS system does not maintain UNIX style permission codes for conventional data sets, the z/OS NFS server returns a code indicating that anyone can access the file. This is done since passing any lesser access code to the client would result in the client user not being allowed to use the cached data which was already read. When the file data is cached on the z/OS NFS client system and another client user on this system attempts to access the same file data, the z/OS NFS client checks the returned permission codes to validate access. Since the z/OS NFS server has passed a code which allows anyone access to the file, all users on the client system can

access the cached data without further restrictions. If data caching is turned off, no client caching takes place and each user must pass the server security check.

Based on the installation time out values, the file data cached by the client is flushed and further attempts to access the file data again requires passing server authorization.

The installation **DataCaching** parameter can be set and it can be overridden for each mount point so that different mount points can be handled as required for the files under that mount point.

If the potential security exposure can not be tolerated for sensitive file data, the **DataCaching** should be used so that no file data is cached by the z/OS NFS client.

Mount processing parameters and installation parameters

Table 18 shows mount processing parameters.

Table 18. Mount processing parameters

acdirmax(n)	proto(tcp udp)
acdirmin(n)	ReadAhead(n)
acregmax(n)	retrans(n)
acregmin(n)	retry(n)
AttrCaching(Y N)	rsize(n)
cln_ccsid(n)	srv_ccsid(n)
DataCaching(Y N)	timeo(n)
DelayWrite(n)	vers(2 3)
delim (binary nl cr lf crlf lfcr)	wsiz(n)
DynamicSizeAdj(Y N)	xlat(Y N)
hard soft	

Table 19 shows installation parameters.

Table 19. Installation parameters

AttrCaching(Y N)	delim (binary nl cr lf crlf lfcr)
Biod(n)	DynamicSizeAdj(Y N)
Bufhigh(n)	ReadAhead(n)
cln_ccsid(n)	srv_ccsid(n)
DataCaching(Y N)	xlat(Y N)
DelayWrite(n)	

The following conditions may cause the NFS client to fail its initialization:

- The NFS client is not started in a standalone colony address space.
- The NFS client is already started; multiple instances of the NFS client on a single multiple virtual system (MVS) is not supported.
- Invalid parameter is specified in the installation parameters.
- If Unicode exists, then Unicode is used. If Unicode does not exist and Character Data Representation Architecture (CDRA) exists, then CDRA is used. If both Unicode and CDRA do not exist, then initialization fails.

A WTO message is issued to the operator console if the NFS client fails to initialize.

Chapter 5. Operating the Network File System

This chapter describes how to start and stop the NFS, and describes the operands of the **modify** command that are related to the z/OS NFS server. The operands for collecting diagnostic information are also described.

Starting the z/OS NFS client

If you want to use the z/OS NFS client, FMID HDZ11TC, use this startup order:

1. Start the z/OS UNIX address space. Wait until this message appears before proceeding: BPXI004I OMVS INITIALIZATION COMPLETE. See *z/OS UNIX System Services Planning* for additional information.
2. Start TCP/IP and **Portmap**. Wait until this message appears: MVPOED0001I OPENEDITION-TCP/IP connection established.

During z/OS UNIX file system initialization, the z/OS NFS client is started and run in the LFS colony address space. The FILESYSTYPE parmlib statement for the z/OS NFS client must be present in the SYS1.PARMLIB(BPXPRMxx) parmlib member in order to start the z/OS NFS client. For more information on z/OS UNIX file system, see *z/OS UNIX System Services File System Interface Reference*.

If the z/OS NFS client fails to initialize, a write to operator (WTO) message is issued to the operator console. The following conditions can cause z/OS NFS client initialization to fail:

- The z/OS NFS client is not started in a standalone colony address space.
- The z/OS NFS client is already started. Multiple instances of the z/OS NFS client on a single MVS system is not supported.
- Using a security product that is downlevel, the z/OS NFS client requires RACF 2.2.
- An incorrect parameter is specified in the installation parameters.
- Character data representation architecture (CDRA) is not installed or is not available.

When the z/OS NFS client initializes, messages like these example messages are displayed on the operator's console:

```
$HASP373 NFSCR   STARTED
BPXI004I OMVS INITIALIZATION COMPLETE
GFSC700I z/OS NETWORK FILE SYSTEM CLIENT
(HDZ11TC) STARTED
```

Stopping the z/OS NFS client

The z/OS NFS client is started when the z/OS UNIX file system is initialized and is persistent until z/OS UNIX is stopped. To stop the z/OS NFS client gracefully the system operator can use the modify operator command **stop** with the z/OS NFS client address space name; for example:

```
stop mvsnfsc
```

When the **stop** command fails to gracefully shut down the z/OS NFS client, the operator can force an abnormal termination using the operator command **cancel** with the z/OS NFS client address space name; for example:

```
cancel mvsnfsc
```

It is imperative and necessary to stop the z/OS NFS client gracefully so it can save its important data (RPC transaction ID) for the subsequent restart.

Attention: The destruction of the z/OS NFS client address space can cause unpredictable abnormal OMVS address space behavior. If a z/OS UNIX process tries to access the NFS client data during its address space destruction, then an OC4 protection exception in the z/OS UNIX BPXVCLNY load module can occur.

To bring down the NFS client during shutdown, follow these steps:

1. Cancel all colony address spaces.
2. The NFS client runs in a colony address space; to terminate it, you can enter either:

```
STOP <nfsc> .....where <nfsc> is the nfs client proc name
```

or

```
CANCEL <nfsc> .....if STOP did not work
```

3. Terminate UNIX System Services.
 - You can use the BPXSTOP tool from the tools & toys section of the UNIX System Services web site (<http://www.s390.ibm.com/oe/bpxa1ty2.html>)
 - Or you can also use:

```
F BPXOINIT,SHUTDOWN=FORKINIT
```

4. Stop JES2.

Starting the z/OS NFS server

Make sure that OMVS is customized to be able to start automatically during IPL.

If you want to use the z/OS NFS server, FMID HDZ11TS, use this startup order:

1. Start TCP/IP and **Portmap**. Wait until the message EZZ4202I or EZY2140I appears:

```
OpenEdition-TCP/IP connection established
```

2. Start the z/OS NFS server.

You might also need to start up NAMESRV to map machine names into their corresponding Internet addresses (NAMESRV is optional).

Note: Portmap is synonymous with Portmapper which is a program provided by MVS TCP/IP that maps registered server programs to port numbers. NAMESRV is the cataloged procedure of the Domain Name Server which is provided by MVS TCP/IP that maps a host name to an internet address or an internet address to a host name. See *TCP/IP for MVS: Customization and Administration Guide* for

information on configuring **Portmap**, starting **Portmap**, configuring the Domain Name Server, and starting NAMESRV automatically with MVS TCP/IP.

The z/OS NFS server does not support file persistence. That is, when the server is restarted, files cannot be accessed using old file handles.

To start the z/OS NFS server, enter the **start** command from a console. Enter the command as follows:

```
▶▶—START—mvsnfs—▶▶
      |
      |┌ info
      |├ error
      |└ warn
      |┌, ,parms='
      |└,altsym'
```

- **mvsnfs** is the name of the procedure in your system that PROCLIB used to start up the server.
- The first parameter specifies the level for diagnostic messages (the default is **info**).
- If **altsym** is specified in the second parameter, the semicolon (;) is used as the comment symbol in the attributes and exports data sets. Otherwise, the pound sign (#) is used as the comment symbol.

For example, suppose you have some data sets with a high-level qualifier of #USER05, and you want client users to be able to read them. First, you would modify the exports data set and attributes data set by using ';' as the comment symbol rather than '#'. Next you would specify #user05 -ro as an entry in the exports data set. Finally, you would specify the **altsym** parameter when you enter the **start** command.

These parameters override the parameter settings in the server startup procedure.

When you enter **start**, this console message appears, if you installed HDZ11TS, then this FMID is displayed in the GFSA328I message text:

```
GFSA348I z/OS NETWORK FILE SYSTEM SERVER
(HDZ11TS) STARTED.
```

Stopping the z/OS NFS server

When shutting down your system, follow these sequential steps:

1. Stop the z/OS NFS server.
2. Sshut down the TCP/IP server.
3. Shut down z/OS UNIX if it is running.

Use the **stop** command to shut down the z/OS NFS server. All current input/output (I/O) operations are completed and all OPEN data sets are closed. This command can be entered at any time. Enter the command as follows, where *mvsnfs* is the name of the procedure in your system that PROCLIB used to start up the server:

```
▶▶—STOP—mvsnfs—▶▶
```

You can also shut down the server by entering the **modify** command:

```
►► MODIFY mvsnfs,stop ◀◀  
  F
```

The operator's console displays messages like these:

```
GFS329I SERVER SHUTDOWN IN PROGRESS.  
GFS330I SERVER SHUTDOWN COMPLETE.  
$HASP395 MVS NFS ENDED
```

As a last resort, you can cancel the server by entering the **cancel** command:

```
►► CANCEL mvsnfs ◀◀
```

Starting the z/OS NFS NSM and z/OS NFS NLM

If you want to use the z/OS NFS Network Status Monitor (z/OS NFS NSM) and the z/OS NFS Network Lock Manager (z/OS NFS NLM), use this startup order.

1. Startup z/OS NFS server ("Starting the z/OS NFS server" on page 66)
2. Startup z/OS NFS NSM
3. Startup z/OS NFS NLM

Notes:

1. The z/OS NFS NSM must be started before the z/OS NFS NLM.
2. You need to define the z/OS NFS NLM and the z/OS NFS NSM Startup Procedures to an z/OS UNIX segment for each RACF as UID(0). For example:

```
ADDUSER (MVSLOCKD) OMVS(UID(0))  
ADDUSER (MVSSTATD) OMVS(UID(0))
```

Starting the z/OS NFS NSM

To start the z/OS NFS NSM server, enter the **start** command from a console. Enter the command as follows:

```
►► START mvsstatd ◀◀
```

mvsstatd is the name of the procedure in your system that PROCLIB used to start up the z/OS NFS NSM.

Note: Do not start the z/OS NFS NLM until you see this FMID message displayed.

```
GFS345I z/OS NETWORK FILE SYSTEM STATUS MONITOR (HDZ11TS)  
STARTED
```

Starting the z/OS NFS NLM

To start the z/OS NFS NLM server, enter the **start** command from a console. Enter the command as follows:

▶▶—START—*mvslockd*—▶▶

mvslockd is the name of the procedure in your system that PROCLIB used to start up the z/OS NFS NLM.

Notes:

1. You will notice a 45 second delay to bring up MVSLOCKD which is the grace period to allow MVSLOCKD to re-establish existing locks.
2. You must see this FMID message before you can use the z/OS NFS NLM.

```
GFS344I z/OS NETWORK FILE SYSTEM LOCK MANAGER (HDZ11TS)
STARTED
```

Stopping the z/OS NFS NSM and the z/OS NFS NLM

When shutting down your system, follow these steps:

1. Stop the z/OS NFS NLM.
2. Shut down the z/OS NFS NSM.

Use the **stop** command to shut down the z/OS NFS NLM. Enter the command as follows, where *mvslockd* is the name of the procedure in your system that PROCLIB used to start up the server:

▶▶—STOP—*mvslockd*—▶▶

Use the **stop** command to shut down the z/OS NFS NSM. Enter the command as follows, where *mvsstatd* is the name of the procedure in your system that PROCLIB used to start up the server:

▶▶—STOP—*mvsstatd*—▶▶

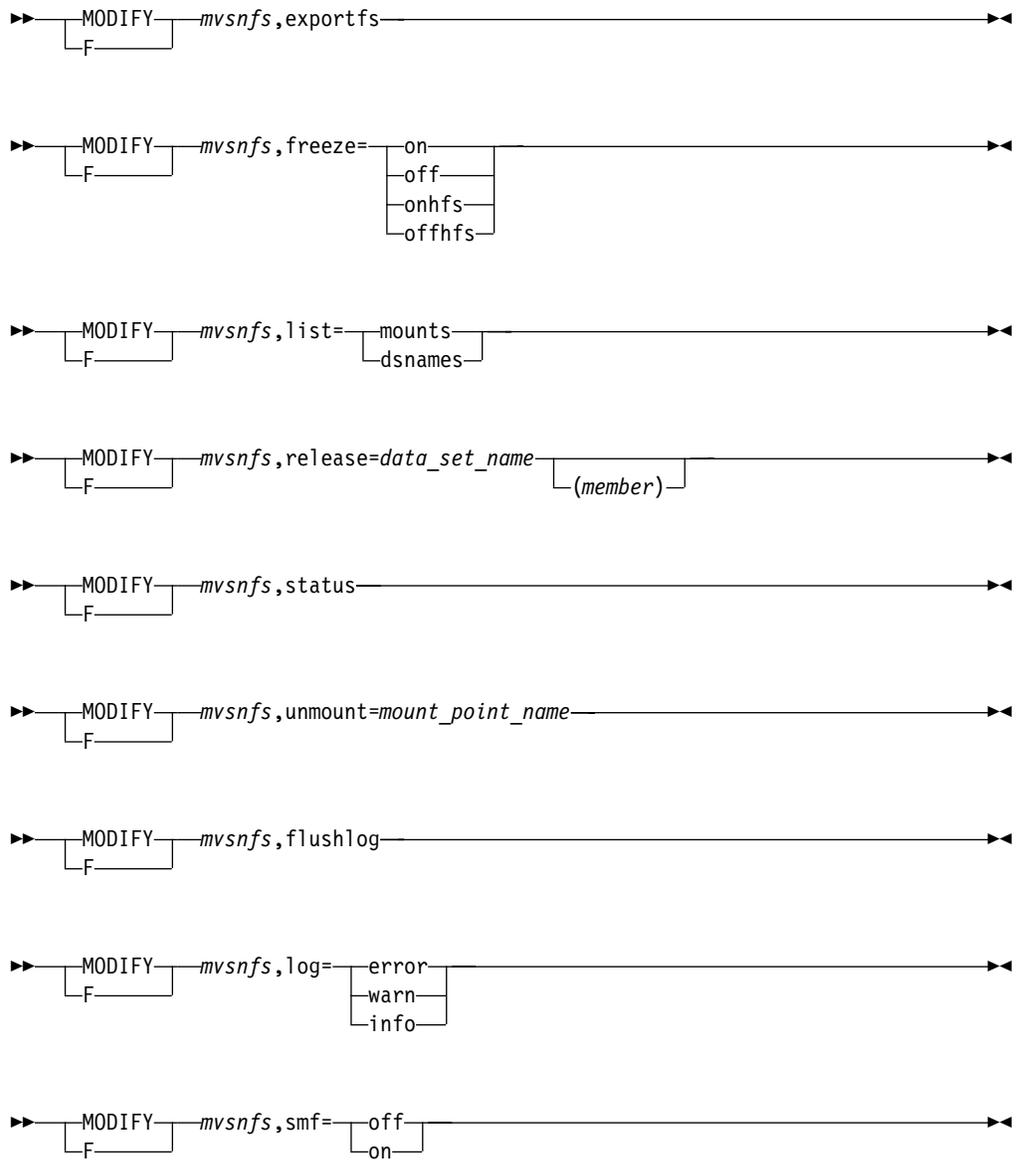
Operands of the modify command for the z/OS NFS server

Besides the **start** and **stop** commands, you can enter the **modify** command with these operands at the console: *exportfs*, *freeze*, *list*, *release*, *status*, *unmount*, *flushlog*, *log*, and *smf*. The *flushlog*, *log*, and *smf* operands can be used to collect data when diagnosing problems. All operands must be preceded by either **MODIFY** *mvsnfs*, or **F** *mvsnfs*, where *mvsnfs* is the name of the procedure in your system that PROCLIB used to start up the server.

Each message is sent to the console or the data set that is pointed to by the NFSLOG1 and NFSLOG2 DD statements. NFSLOG1 and NFSLOG2 are DD statements in the startup procedure of the z/OS NFS server. These z/OS NFS server DD statements specify data sets where all the messages for debugging or trace are stored during the processing of the z/OS NFS server.

All data set names entered from the console must be fully-qualified and without quotation marks.

This is a summary of the **modify** command with the relevant operands:



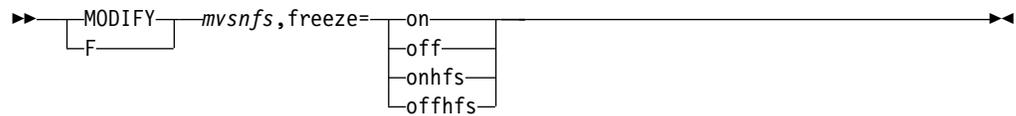
Exportfs operand

This operand causes the exports data set to be re-read and the internal exports list to be rebuilt without stopping and restarting the server. This operand can be run at any time. Enter the operand as follows:



Freeze operand

This operand suspends or resumes processing of user mount requests. You can enter the *freeze=on* command at any time for maintenance purposes. Enter the operand as follows:



- If you select **on**, these messages appear:

```
GFS901I MOUNT PROCESSING SUSPENDED.
GFS971I OPENEDITION HFS MOUNTS SUSPENDED.
```

Future mount requests by client users are refused for both HFS file systems and conventional MVS data sets and the message "Permission Denied" displays on their monitors. After issuing a *freeze=off* operand, mount processing resumes for both HFS file systems and conventional MVS data sets, and these messages appear:

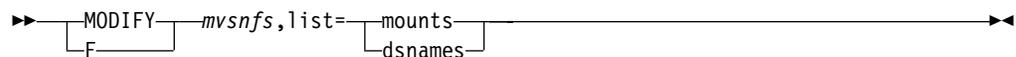
```
GFS902I MOUNT PROCESSING RESUMED.
GFS972I OPENEDITION HFS MOUNTS RESUMED.
```

Client users can again mount MVS directories as normal.

- If you select **onhfs**, future mount requests by client users for HFS file systems are refused.
- If you select **offhfs**, mount processing resumes for HFS file systems.

List operand

This operand displays a list of either the mount points or the MVS data sets that are currently active in the Network File System. This command can be entered at any time for maintenance or diagnostic purposes. The path name output of an HFS file object might require multiple console messages. Enter the operand as follows:



The **mounts** option returns a list of all mount points currently active in the system. The list of active mount point names and their associated *current use counts* like these are displayed:

```
GFS910I JOHN ACTIVE =12.
GFS910I JOAN ACTIVE =10.
GFS910I /HFS/U/BLUE ACTIVE =1.
```

Current use counts indicate how many mount requests have been made without an unmount request for the same mount point regardless to which local directory the mount is attached. For example, suppose the high-level qualifier JOHN is mounted to the same local directory 12 times without any unmount. ACTIVE =12 is shown. Now, suppose the high-level qualifier JOAN has been mounted to 15 different local directories but 5 of them have been unmounted. ACTIVE =10 is shown.

The **dsnames** option returns a list of all data sets that are currently active. Currently active means that the data set is either in use or has been opened for

accessing but has not been closed due to timeout even though the file is not in use. A list of data sets and data set members such as these are displayed:

```
GFSA912I CHRIS.TEST.  
GFSA912I SMITH.PAYROLL(JULY).  
GFSA912I /HFS/U/PAYROLL.
```

HFS file names are truncated after 126 characters.

Release operand

This operand forces the Network File System to release a data set or a member of a PDS. If the data set is active, the server closes and deallocates it. For HFS file objects, this command is ignored. This command can be entered at any time. Enter the operand as follows:

```
►►┌──┴──┐ MODIFY ─ mvsnfs,release=data_set_name ─┬──┴──►►  
└──┬──┘ └──┬──┘  
F (member)
```

- If the data set you specify is active, you receive a message as follows:

```
GFSA914I data.set.name(member) DEALLOCATED.
```

- If the data set you specify is not active, you receive a message as follows:

```
GFSA915I TINA.WORKLIB.REVENUE(AREA4) NOT ALLOCATED.
```

Status operand

This operand displays the status of the server's active subtasks. You can enter this command for diagnosis purposes at any time that the server is running. Enter the operand as follows:

```
►►┌──┴──┐ MODIFY ─ mvsnfs,status ─┬──┴──►►  
└──┬──┘  
F
```

This is a sample message listing:

```
GFSA900I MOUNT PROCESSING ACTIVE.  
GFSA751I SMF PROCESSING SUSPENDED FOR USER LOGOUT.  
GFSA753I SMF PROCESSING SUSPENDED FOR FILE TIMEOUT.  
GFSA781I OPENEDITION SAF PROCESSING ENABLED.  
GFSA903I TASK 5C580 TCB 8D1888 PROGRAM = GFSASUBT =  
NFSTSK02  
GFSA903I TASK 5C7A0 TCB 8D10C8 PROGRAM = GFSASUBT =  
NFSTSK01  
GFSA903I TASK 5C9C0 TCB 8D1378 PROGRAM = GFSASUBT =  
NFSTSK00  
GFSA903I TASK 5CE00 TCB 8D1378 PROGRAM = GFSAXPRT =  
TRANSPORT
```

Unmount operand

This operand unmounts a mount point that is currently active. The path name for an HFS file object cannot exceed 126 bytes. The mount_point_name for an HFS path must be specified in single quotes with the hfs prefix (/HFS) in uppercase followed

by the `hfs` path (case sensitive). For example: `F mvsnfs,unmount='/HFS/u/jones'`. The general command syntax is as follows:

```

  >> [MODIFY] mvsnfs,unmount='mount_point_name' <<<
      |
      | F
  
```

- If the mount point is active, the server responds:

```
GFSA916I mount_point UNMOUNTED.
```

- If the mount point you specify does not exist when you enter the command, the server displays messages similar to these:

```
GFSA917I SMITH NOT MOUNTED.
GFSA917I /HFS/U/JONES NOT MOUNTED.
```

Once the mount point is removed, client users are unable to access this mount point, and they get the “Stale NFS File Handle” message. Client users have to enter the **umount** command to end the stale file handle problem.

Operands of the modify command for diagnosis

These operands of the **modify** command can help you to collect data for diagnosing problems.

Note: All operands must be preceded with either:

```
MODIFY mvsnfs,
```

or

```
F mvsnfs,
```

where `mvsnfs` is the name of the procedure in your system that PROCLIB used to start up the server.

Each message is sent to the console or the data set that is pointed to by the NFSLOG1 or NFSLOG2 DD statements.

Flushlog operand

flushlog is an operator command that lets you flush the NFS message log to disk. This command enables a TSO user to browse all the log records that have been written by the NFS. Enter the operand as follows:

```

  >> [MODIFY] mvsnfs,flushlog <<<
      |
      | F
  
```

Log=msglevel operand

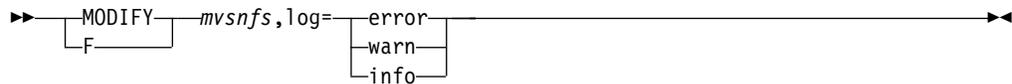
Use this operand to set the level of NFS logging messages to be collected:

log=error - Collects only error messages

log=warn - Collects error message and attention messages

log=info - Collects error, attention, and informational messages

Enter the operand as follows:



The diagnostic message level can also be set in the execution statement of the Network File System startup procedure.

For more information, see “Log data sets” on page 95.

Smf operand

When the z/OS NFS server starts up, *smf* is set to off. You need to set *smf* to on if you want to collect the SMF records. The Network File System suspends writing SMF records when it detects an SMF error. The *smf* operand is used to resume or suspend the collection of SMF data. This operand has two parameters, ON and OFF.

smf=on - Resumes the collection of SMF records

smf=off - Suspends the collection of SMF records

Enter the operand as follows:



Chapter 6. Network File System installation-wide exits for the z/OS NFS server

This chapter helps you write installation-wide exits or replaceable modules that customize NFS processing. It contains product-sensitive programming interfaces that are provided by the server.

If there is no customization made to the installation-wide exit routines, we recommend that you remove them for better performance.

Requirements for the NFS

Table 20 summarizes the installation-wide exit routines that are provided by the NFS.

Table 20. List of installation-wide exits

Source Module Name	Load Module Name	Parameter List Mapping Macro	Description
GFSAXL	GFSAXL	GFSAXLOG GFSAXDSA	Login security exit
GFSAXF	GFSAXF	GFSAXSEC GFSAXDSA	File security exit

GFSAXDSA is a sample skeleton for a user storage block.

The above exits are shipped with the NFS and they contain the dummy skeleton code. The source modules reside in the *prefix.NFSSAMP* data set and the macros reside in *prefix.NFSMAC* data set. (The value of *prefix* depends on the installation.) Before modifying or replacing these exits, you should review the functions and processing of these exit routines carefully. These are basic requirements for all the NFS exit routines:

- Exit routines must reside in an authorized program library and be accessible by the MVS LOAD macro. NFS and installation-wide exits receives control in problem state key 8. Installation-wide exits are run as an APF-authorized task, because Network File System is APF-authorized. As with any APF-authorized program, your exits should not be link-edited with APF-authorization. Only the main task, NFS, should have that link edit attribute.
- Exit routines must be link-edited with AMODE(31).
The installation-wide exits are entered in AMODE(31) and can reside above or below the 16M line depending on the requirements of the installation-wide exits themselves.
- Exit routines must be reentrant.
- Exit routines must follow the standard MVS register save and restore convention. The standard MVS registers convention is:
 - Register 1 contains the address of the exit parameter list.
 - Register 13 contains the address of the caller's save area.
 - Register 14 contains the caller's return address.
 - Register 15 contains the address of the entry point for this exit routine. The server does not use return codes stored in register 15, but includes a parameter in the parameter list for exits that supply return codes.

Notes:

1. Address parameters have null value (0) if the related data does not exist.
2. The length of each field can be found in the macros shipped. Field length can be changed in the future.

Sample link-edit JCL

Use the sample JCL shown in Figure 9 to assemble and link-edit the GFS AUXL and GFS AUXF load modules:

```
//jobname JOB (job_and_user_accounting_information)
//EXITASM PROC M=
//ASM EXEC PGM=ASMA90,
// PARM='RENT'
//SYSPRINT DD SYSOUT=*
//SYSIN DD DISP=SHR,DSN=source_library_name(&M)
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
// DD DISP=SHR,DSN=source_macro_library_name
//SYSLIN DD DISP=SHR,DSN=obj_library_name(&M)
//SYSUT1 DD UNIT=SYSDA,
// SPACE=(32000,(30,30))
//PEND
// EXEC EXITASM,M=GFS AUXL
// EXEC EXITASM,M=GFS AUXF
//stepname EXEC PGM=HEWL,
// PARM='MAP,LIST,RENT,REUS'
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLMOD DD DSN=apf_library_name,DISP=OLD
//USER DD DSN=obj_library_name,DISP=SHR
//SYSLIN DD *
INCLUDE USER(GFS AUXL)
MODE AMODE(31)
ENTRY GFS AUXL
NAME GFS AUXL(R)
INCLUDE USER(GFS AUXF)
MODE AMODE(31)
ENTRY GFS AUXF
NAME GFS AUXF(R)
/*
```

Figure 9. Sample link-edit JCL for the NFS. This example uses High Level Assembler. To use Assembler H, replace ASMA90 on the EXEC statement with IEV90. The rest of the JCL would be the same.

Using storage blocks of the server exits

This section discusses how to use storage blocks of the NFS installation-wide exits.

Global exit block (GXB)

The GXB is obtained once during system initialization by the server *login* exit. The exit returns a word to the server. This word is referred to as 'the address of the GXB', but the system initialization exit might store any value in the word. The address of the GXB is returned to the server in the parameter list and passed back to the installation-wide exits in each subsequent call. This block contains user installation-wide exit data that is needed to communicate with the server.

The GXB can contain an area to save the user data needed for all sessions. The usage of this block is determined by the exit. Access to the Global Exit Block must be controlled by the user-written installation-wide exits to ensure that updates are serialized and do not interfere with each other. This block is shared with the *file*

security Installation-Wide Exit. The format of the GXB is entirely controlled by the login and file security installation-wide exits.

User exit block (UXB)

During a *Start of New User Session* request or a *User Login Request*, the exit can obtain a User Exit Block. The exit returns a word to the server. This word is referred to as 'the address of the UXB', but the system initialization exit might store any value in the word. The address of the UXB is returned to the server in the parameter list (depends on which request comes first), and is passed back to the installation-wide exits on each subsequent call related to this combination of machine and user IDs.

The UXB can contain an area to save the user data needed for this session. The usage of this block is determined by the exit. The exit is responsible for obtaining, and freeing access to these storage areas. This block is used by the *login* installation-wide exit and *file security* installation-wide exit. The format of the UXB is entirely controlled by the login and file security installation-wide exits.

Login installation-wide exit

The exit routine can invoke a customized authorization facility. The server mainline code can be set to perform Security Authorization Facility (SAF) checking, by specifying the **security** attribute in the attributes table.

If **security(saf)** or **security(safexp)** is specified in the attributes table and the exit routines exist, these exit routines get control first, and then SAF security checking gets control. If the exit routines fail the request, the entire request fails. If the exit routines process the request successfully, then the request is processed by the SAF checking. Similarly, if the SAF checking fails the request, the entire request fails.

If neither **security(saf)** nor **security(safexp)** is specified in the attributes table and the exit routine exists, this exit routine determines whether the request is successful or fails.

Figure 10 on page 78 shows the logic flow that determines which login checking routines are used.

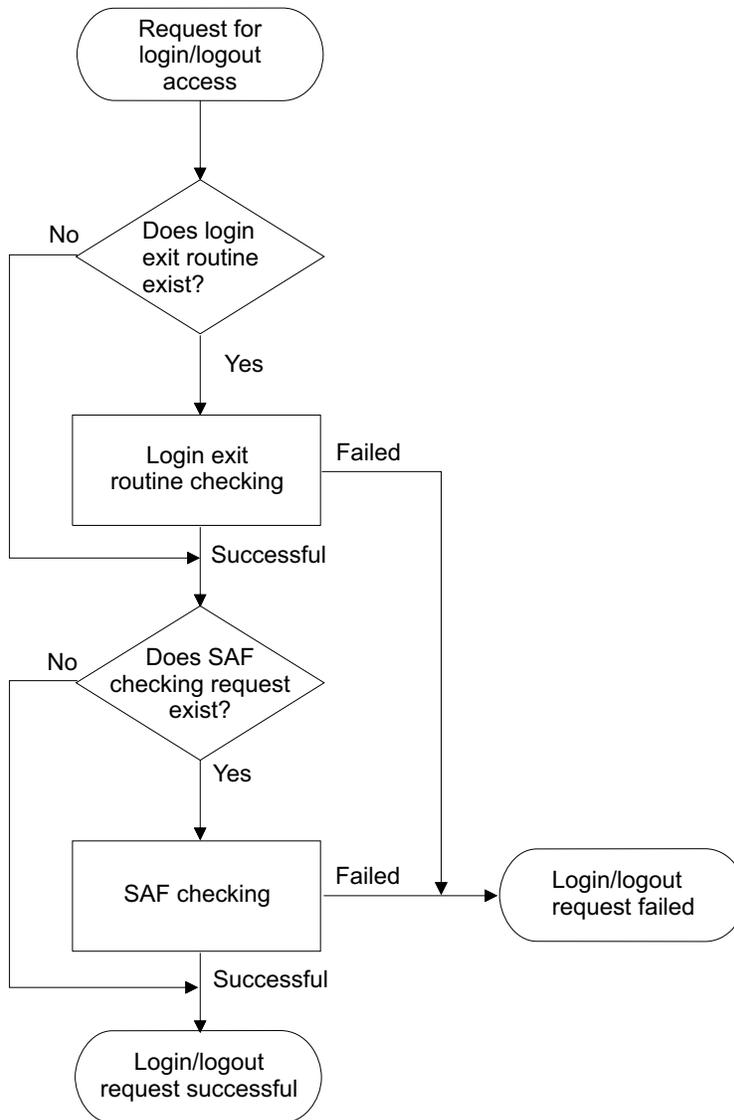


Figure 10. Determining which login checking routines are used

The *login* installation-wide exit has a parameter list which is passed from the server to the installation-wide exit. The *login* installation-wide exit can be invoked for any of the following conditions:

- **System initialization** - Performed once during the initialization of the server and allows a Global Exit Block to be obtained. The GXB address is always returned to the installation-wide exits (see GXB in “Requirements of the login exit” on page 79). If this request fails, both the login installation-wide exit and the file security installation-wide exit are marked as non-existent.
- **New user session** - Performed when a unique combination of UNIX UID and Internet address is detected. The exit might obtain a User Exit Block for use by later calls if the UXB does not exist.
- **Login** - Performs user verification when a client tries to use either an **mvslogin** command or a PCNFSD request. The exit might obtain a User Exit Block for use by later calls if the UXB does not exist.
- **Logout** - Performs cleanup when a client tries to use the **mvslogout** (or **mvslogut**) command or timeout occurs. Timeout is the value specified in the

logout attribute in the attributes table. On a logout, the UXB is released. Logout can also be initiated by the login request.

- **System termination** - Occurs once during the termination of the server, and causes the Global Exit Block to be freed.

Requirements of the login exit

Besides the requirements stated in “Requirements for the NFS” on page 75, the *login* installation-wide exit routine must be link-edited with the name of GFSAXL.

Options of the login exit

The *login* installation-wide exit routine can perform these functions:

- Obtain a Global Exit Block (GXB)
- Obtain a User Exit Block (UXB)

If the User Exit Block already exists, its address is passed to the installation-wide exit routine; otherwise, 0 is passed. The installation-wide exit routine can accept or reject the request. If the request is accepted and the User Exit Block does not exist, you can allocate the User Exit Block and return the address in the LEDXSX parameter list.

Structure of the login exit message

The message supplied by the login exit is sent to the NFS server log data set. The message has the label name LEDSX and consists of these two fields:

LEDXCNT - 1 byte message length, excluding the NULL character (X'00')

LEDXMGS - 81 byte NULL character string. The installation-wide exit can fill in this message field with the message ended by one or more bytes of X'00'

If the format of the message is incorrect, both the login installation-wide exit and the file security installation-wide exit are marked as nonexistent, and a message (GFA960I or GFA961I) is sent to the z/OS NFS server log data set and to the console.

Contents of the login exit parameter list

The parameter list is mapped by macro and DSECT GFSAXLOG. Table 21 describes the contents of each field.

Table 21. Format of login installation-wide exit routine parameter list

Field Name	Description	Contents
LEDSRQ	Request code	System request code set by the server before calling this installation-wide exit, for these conditions: 4 System initialization 8 System termination 12 Start of new user Session 16 User login request 20 Logout has been requested
LEDSRC	Return code	Codes generated and returned by the calls.
LEDISM	Client machine name	Character string ended by single byte containing X'00'.

Table 21. Format of login installation-wide exit routine parameter list (continued)

Field Name	Description	Contents
LEDSIA	Client IP address	Number (32-bit Internet address)
LEDSU	UNIX Client user ID number	Number
LEDSG	UNIX Client group ID number	Number
LEDSMU	MVS user ID	Character string padded with a blank at the end of the user ID, conforming to MVS standards.
LEDSMG	MVS group name	Character string padded with a blank at the end of the group name, conforming to MVS standards.
LEDSXS	Address of UXB	Size and content are installation-dependent, generated at the start of a new user session or a user login request.
LEDSXG	Address of GXB	Size and content are installation-dependent, generated at system initialization.
LEDSXD	Message supplied by this exit routine	Message structure (see "Structure of the login exit message" on page 79).

Note: Character strings are in upper-case EBCDIC.

Using the login exit parameter list

This section describes how the Global Exit Block (GXB) or User Exit Block (UXB) is constructed and used between the *login* installation-wide exit and the NFS. A request code is set by the server before each call to this installation-wide exit routine. The installation-wide exit routine provides a return code for each event.

Request codes to the login exit

Table 22 shows the *login* installation-wide exit request codes and their meanings:

Table 22. Request codes to the login exit

Code	Meaning
4 (X'04')	System initialization
8 (X'08')	System termination
12 (X'0C')	Start of new user session
16 (X'10')	User login request
20 (X'14')	User logout request

IBM might add new request codes in a future level of the server. To provide for this, consider making your exit set a return code 0 if it does not recognize the request code.

Return codes from the login exit

Table 23 on page 81 shows the *login* installation-wide exit routine return codes and their meanings:

Table 23. Return codes from the login exit

Code	Meaning
0 (X'00')	Request successful
4 (X'04')	Request unsuccessful

System initialization routine of the login exit

The system initialization routine is called each time the server address space starts, and can acquire and initialize the GXB. Table 24 shows the codes and fields that are used for this routine:

Table 24. Codes and fields for system initialization

When Invoked	Field Name	Description	Contents
On entry	LEDSRQ	Request code	4
On exit	LEDSXD	Exit-supplied error message	Message Structure
On exit	LEDSRC	Return code	0 Initialization successful 4 Stop the NFS.
On exit	LEDSXG	Global Exit Block	Address

Start of new user session routine of the login exit

The start of new user session routine is called when a new client machine-user combination is recognized by the server. The exit might acquire and initialize a UXB. Table 25 shows the codes and fields that are used for this routine:

Table 25. Codes and fields for start of new user session

When Invoked	Field Name	Description	Contents
On entry	LEDSRQ	Request code	12
On entry	LEDSM	Client machine name	EBCDIC character string
On entry	LEDSIA	Client IP address	Number
On entry	LEDSU	UNIX Client User ID number	Number
On entry	LEDSG	UNIX Client Group ID number	Number
On entry	LEDSXS	User Exit Block	0
On entry	LEDSXG	Global Exit Block	Value
On exit	LEDSXD	Exit-supplied error message	Message Structure
On exit	LEDSRC	Return code	0 New session established 4 New session not established

Table 25. Codes and fields for start of new user session (continued)

When Invoked	Field Name	Description	Contents
On exit	LEDSXS	User Exit Block	Value, if return code in LEDSRC is 0

User login request routine of the login exit

This routine is called when the **mvslogin** command or PCNFSD request is used. The installation security system should be called to determine if the user is properly authorized. Table 26 shows the codes and fields:

Table 26. Codes and fields for user login request

When Invoked	Field Name	Description	Contents
On entry	LEDSRQ	Request code	16
On entry	LEDSM	Client machine name	Character string
On entry	LEDSIA	Client IP address	Number
On entry	LEDSU	UNIX Client User ID number	Number
On entry	LEDSG	UNIX Client Group ID number	Number
On entry	LEDSMU	MVS User ID	Character string
On entry	LEDSMG	MVS Group Name	Character string
On entry	LEDSXS	User Exit Block	Address or 0
On entry	LEDSXG	Global Exit Block	Address
On exit	LEDSXD	Exit-supplied error message	Message Structure
On exit	LEDSRC	Return code	0 <i>Login successful</i> 4 <i>Login failed</i>
On exit	LEDSXS	User Exit Block	Value

User logout request routine of the login exit

This routine is used at logout to return the User Exit Block storage obtained at the start of the session and to perform any related logout processing. The codes and fields used are shown in Table 27.

Table 27. Codes and fields for logout request

When Invoked	Field Name	Description	Contents
On entry	LEDSRQ	Request code	20
On entry	LEDSM	Client machine name	Character string
On entry	LEDSIA	Client IP address	Number
On entry	LEDSU	UNIX Client User ID number	Number
On entry	LEDSG	UNIX Client Group ID number	Number
On entry	LEDSMU	MVS User ID	Character string

Table 27. Codes and fields for logout request (continued)

When Invoked	Field Name	Description	Contents	
On entry	LEDSMG	MVS Group Name	Character string	
On entry	LEDSXS	User Exit Block	Address or 0	
On entry	LEDSXG	Global Exit Block	Address	
On exit	LEDSXD	Exit-supplied error message	Message Structure	
On exit	LEDSRC	Return code	0	Logout successful
On exit	LEDSXS	User Exit Block	0	UXB is released

System termination routine of the login exit

This routine is used at server termination to release the GXB storage. All users are automatically logged off. The codes and fields used are shown in Table 28.

Table 28. Codes and fields for system termination

When Invoked	Field Name	Description	Contents	
On entry	LEDSRQ	Request code	8	
On entry	LEDSXG	Global Exit Block	Address	
On exit	LEDSXD	Exit-supplied error message	Message Structure	
On exit	LEDSRC	Return code	0	Exit termination successful
On exit	LEDSXG	Global Exit Block	0	GXB is released

File security installation-wide exit

The file security installation-wide exit routine verifies that a user is authorized to access a data set or data set member with the access mode requested. If the request from allocation, write, read, or access does not have permissions set up, then the exit routine gets control.

The permissions set up by the file security exit can be overridden by the SAF checking. If the exits allow access and there is no SAF checking, the permissions remain in effect until logout. The server does not call again for the same access before logout. The server gets the access mode or permissions before any of the other three types of calls.

If **security(saf)** or **security(safexp)** is specified in the attributes table and the exit routine exists, this exit routine gets control first, and then SAF security checking gets control. If the exit routines fail the request, the entire request fails. If the exit routines process the request successfully, then the request is processed by the SAF

checking. Similarly, if the SAF checking fails the request, the entire request fails. If the SAF checking is successful, the file permissions from the SAF checking are set up for the request.

If neither **security(saf)** nor **security(safexp)** is specified in the attributes table and the exit routine exists, this exit routine determines the permissions.

Figure 11 shows the logic flow determining which file security checking routines are used.

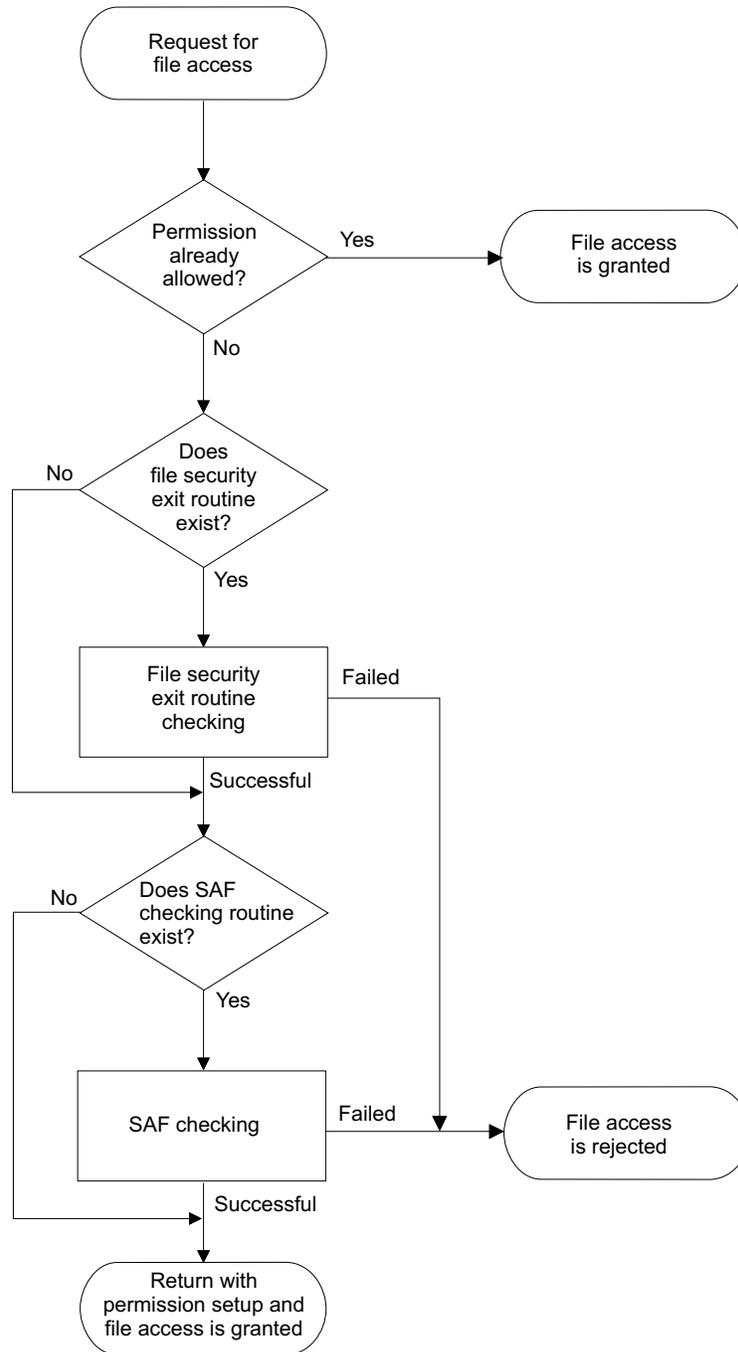


Figure 11. Determining which file security checking routines are used

The file security installation-wide exit uses the parameter list which is used by the server installation-wide exit. This exit is called for at these times:

- **Data Set Allocation** - When a client user creates, renames, or deletes a data set.
- **Data Set Write** - When a client user first writes to an MVS data set or data set member.
- **Data Set Read** - When a logged in user tries to first read from an MVS data set or data set member.
- **Getting Access Mode or Permissions** - This function is called first to set up permission for the read, write, create, delete, or rename data set request. The server needs to obtain the access mode or the permissions that a user has for a data set. This function is only called when SAF is not being used.

The access modes and what they permit are:

Allocate - Read, write, create, delete, or rename the data set

Write - Read or write the data set

Read - Read the data set

A return code is set by the installation-wide exit indicating whether the request is allowed. The file security installation-wide exit is not called at server startup or shutdown.

Requirements of the file security exit

Besides "Requirements for the NFS" on page 75, these requirements must be fulfilled:

- The file security installation exit must be link-edited with the name of GFSAXUF.
- The *login* exit must exist (GFSAXUL).
- If the GXB or UXB are to be used, the *login* exit must obtain them.
- The UXB and GXB are shared with the *login* installation-wide exit.

Structure of the file security exit message

The message supplied by the file security exit is sent to the NFS server log data set. The message has the label name FEDSXD and consists of these two fields:

FEDSXCNT - 1 byte message length, excluding the NULL character (X'00')

FEDSXMGS - 81 byte NULL character string. The installation-wide exit can fill in this message field with an error message ended by one or more bytes of X'00'

If the format of the message is incorrect, the file security installation-wide exit is marked as nonexistent, and a message (GFSAX960I or GFSAX961I) is sent to the NFS server log data set and to the console.

Contents of the file security exit parameter list

The file security installation-wide exit parameter list is mapped by macro and DSECT GFSAXUSEC. Table 29 on page 86 describes each field.

Table 29. Format of the parameter list for the file security installation-wide exit

Field Name	Description	Contents
FEDSRQ	Request code	System request code set by the server before each call to this exit for these conditions: 4 Validate allocate request 8 Validate write request 12 Validate read request 16 Return the current settings of the security permissions
FEDSRC	Return code	Codes are generated and returned by the exit routine.
FEDSM	Client system name	Character string ended by a single byte containing X'00'.
FEDSIA	Client IP address	Number
FEDSU	Client user ID number	Number
FEDSG	Client group ID number	Number
FEDSMU	MVS user ID	Character string padded with a blank at the end of the user ID, conforming to MVS standards.
FEDSDN	MVS data set name	Character string ended by a single byte containing X'00'. This conforms to MVS standards.
FEDSMN	MVS data set member name	Character string ended by a single byte containing X'00'. This conforms to MVS standards.
FEDSXS	Address of UXB	Size and contents are installation dependent. Generated at the start of a user session or user login by the <i>login</i> exit routine.
FEDSXG	Address of GXB	Size and content are installation dependent. Generated at server initialization by the <i>login</i> exit routine.
FEDSTYPE	File Type	Address of fullword integer defining the data set organization: 0 Not used 1 Sequential 2 Partitioned 3 Direct 4 Reserved 5 VSAM (if unable to classify further) 6 VSAM ESDS 7 VSAM RRDS 8 VSAM KSDS 9 Reserved 10 Reserved

Table 29. Format of the parameter list for the file security installation-wide exit (continued)

Field Name	Description	Contents
FEDSRPM	UNIX Permissions	UNIX file modes of type mode_t as defined by POSIX.
FEDSXD	Message supplied by this exit routine	Address of the message structure (see "Structure of the file security exit message" on page 85).

Note: Character strings are in upper-case EBCDIC.

Using the file security exit parameter list

These sections each describe how the data in the parameter list is used by the server and the file security installation-wide exit routine. A request code is set by the server before each call to this installation-wide exit routine.

Each topic below describes an event, for which some fields are set on entry. The file security installation-wide exit provides a return code for each event.

Request codes to the file security exit

Table 30 shows the request codes to the file security installation-wide exit:

Table 30. Request codes to the file security exit

Code	Meaning
4 (X'04')	Validate allocate request
8 (X'08')	Validate write request
12 (X'0C')	Validate read request
16 (X'10')	Return the current settings of the security permissions

Return codes from the file security exit

Table 31 shows the request codes returned by the file security installation-wide exit:

Table 31. Return codes from the file security exit

Code	Meaning
0 (X'00')	Request successful
4 (X'04')	Request unsuccessful

Validate allocate request routine of the file security exit

This routine is called when a user allocates, renames, or deletes a data set or data set member. The codes and fields used are shown in Table 32.

Table 32. Codes and fields for validate allocate request

When Invoked	Field Name	Description	Contents
On entry	FEDSRQ	Request code	4
On entry	FEDSM	Client machine name	Character string
On entry	FEDSIA	Client IP address	Number
On entry	FEDSU	Client User ID number	Number

Table 32. Codes and fields for validate allocate request (continued)

When Invoked	Field Name	Description	Contents	
On entry	FEDSG	Client Group ID number	Number	
On entry	FEDSMU	MVS user ID	Character string	
On entry	FEDSDN	MVS data set name	Character string	
On entry	FEDSMN	MVS data set member name	Character string	
On entry	FEDSXS	User Exit Block	Address	
On entry	FEDSXG	Global Exit Block	Address	
On entry	FEDSTYPE	File type	Number	
On exit	FEDSXD	Exit-supplied error message	Message Structure	
On exit	FEDSRC	Return code	0	Access allowed
			4	Access denied

Validate write request routine of the file security exit

This routine is called when a user writes to an MVS data set or data set member. The codes and fields used are shown in Table 33.

Table 33. Codes and fields for validate write request

When Invoked	Field Name	Description	Contents	
On entry	FEDSRQ	Request code	8	
On entry	FEDSM	Client machine name	Character string	
On entry	FEDSIA	Client IP address	Number	
On entry	FEDSU	Client User ID number	Number	
On entry	FEDSG	Client Group ID number	Number	
On entry	FEDSMU	MVS user ID	Character string	
On entry	FEDSDN	MVS data set name	Character string	
On entry	FEDSMN	MVS data set member name	Character string	
On entry	FEDSXS	User Exit Block	Address	
On entry	FEDSXG	Global Exit Block	Address	
On entry	FEDSTYPE	File type	Number	
On exit	FEDSXD	Exit-supplied error message	Message Structure	
On exit	FEDSRC	Return code	0	Access allowed
			4	Access denied

Validate read request routine of the file security exit

This routine is called when a user reads from an MVS data set or data set member. The codes and fields used are shown in Table 34.

Table 34. Codes and fields for validate read request

When Invoked	Field Name	Description	Contents
On entry	FEDSRQ	Request code	12
On entry	FEDSM	Client machine name	Character string
On entry	FEDSIA	Client IP address	Number
On entry	FEDSU	Client User ID number	Number
On entry	FEDSG	Client Group ID number	Number
On entry	FEDSMU	MVS user ID	Character string
On entry	FEDSDN	MVS data set name	Character string
On entry	FEDSMN	MVS data set member name	Character string
On entry	FEDSXS	User Exit Block	Address
On entry	FEDSXG	Global Exit Block	Address
On entry	FEDSTYPE	File type	Number
On exit	FEDSXD	Exit-supplied error message	Message Structure
On exit	FEDSRC	Return code	0 Access allowed 4 Access denied

Return security permissions routine of the file security exit

This routine is called to query a data set access mode or permission. The codes and fields used are shown in Table 35.

Table 35. Codes and fields for return security permissions

When Invoked	Field Name	Description	Contents
On entry	FEDSRQ	Request code	16
On entry	FEDSM	Client machine name	Character string
On entry	FEDSIA	Client IP address	Number
On entry	FEDSU	Client User ID number	Number
On entry	FEDSG	Client Group ID number	Number
On entry	FEDSMU	MVS user ID	Character string
On entry	FEDSDN	MVS data set name	Character string
On entry	FEDSMN	MVS data set member name	Character string
On entry	FEDSXS	User Exit Block	Address

Table 35. Codes and fields for return security permissions (continued)

When Invoked	Field Name	Description	Contents
On entry	FEDSXG	Global Exit Block	Address
On entry	FEDSTYPE	File type	Number
On exit	FEDSXD	Exit-supplied error message	Message Structure
On exit	FEDSRC	Return code	<p>0 Exit processing successful</p> <p>4 Exit failed</p>
On exit	FEDSRPM	New permissions in mode_t format	UNIX file modes of type mode_t as defined by POSIX

Chapter 7. Diagnosing and reporting problems

This chapter is intended to help you diagnose NFS problems and use keywords to describe the NFS program failures.

Before you begin diagnostic procedures, verify that the suspected problem is not a result of failure caused by:

- MVS TCP/IP (Transmission Control Protocol/Internet Protocol)
- Broken clients (for example, sending incorrect file handles or ignoring the server's error return code)
- The network (for example, packets not arriving at the server, clients not receiving replies, or a gateway going down)
- The result of user input error

The failure caused by MVS TCP/IP, see *TCP/IP for MVS: User's Guide* and *TCP/IP for MVS: Customization and Administration Guide* to diagnose the problem.

Correcting input error

For user input error, use this procedure to assist in correcting the error:

1. Look up the command and its attributes (or parameters), the attributes in the attributes data set, and the entries in the exports data set in this book, in *z/OS Network File System User's Guide*, or the appropriate client machine command documentation.
2. Examine the attributes (or parameters) specified by each command, specified in the attributes data set, and the DIRECTORY entries specified in the exports data set to verify that they are specified correctly. Check the z/OS NFS server log data set, the z/OS NFS client log data set, or both log data sets to find the error message if applicable.
3. If you find a user input error, reenter the command or restart the server after you correct the error.
4. If all attributes (or parameters) and directory statements appear to be specified correctly, you should use this chapter to build a set of keywords that describe the error, and then contact IBM for help.

After determining that the suspected failure is neither an MVS TCP/IP program error nor a user error, you need to develop a *set of keywords* that describe the program failure. A *keyword* is an agreed-upon word or abbreviation used to describe a single aspect of a program failure.

After you have selected a set of keywords, use the set to search IBMLink/Service to determine whether an authorized program analysis report (APAR) has already been recorded for the failure. The IBMLink/Service online database contains information about the resolution of APARs. If a program failure is identified in these databases with the same set of keywords, the search yields a description of the problem and usually a fix. If the failure is not known, use the keywords to describe the failure when you contact IBM for help, which is the final step before an APAR is generated. For more information about these services see *z/OS DFSMSdfp Diagnosis Guide*.

Using keywords to identify the problem

In general, when you contact IBM, you are asked to identify the problem with a full set of keywords. A full set of keywords for the NFS identifies these specific keywords:

- Component identification keyword
- Release level keyword
- Type-of-failure keyword
- Service level keyword

This example displays a full set of keywords:

```
5695DF121 R1TS MSGGFSA865I UW12345
```

Notes:

1. 5695DF121 - Signifies the component identification keyword.
2. R1TS - R1TS is the release level keyword for the z/OS NFS server, FMID HDZ11TS. R1TC is the release level keyword for the z/OS NFS client, FMID HDZ11TC.
3. MSGGFSA865I - MSGGFSA is the type-of-failure keyword prefix for the z/OS NFS server. MSGGFSC is the type-of-failure keyword prefix for the z/OS NFS client.
4. UW12345 - Service level keyword.

These sections explain individual keywords and their relation to the full set of keywords used to describe a NFS program failure.

A search in IBMLink/Service using the first keyword, the NFS component identifier, alone would detect all reported problems for the entire program product. Each keyword added to the search argument makes the search more specific, thereby reducing the number of problem descriptions to be considered. Sometimes you can find a correction for a problem without using the full set of keywords. Each keyword after the first is optional.

Component identification keyword

The component identification number is the first keyword in a set. You should use this keyword whenever the Network File System is suspected of being the failing component. The component identification keyword should be used with at least one other keyword to search IBMLink/Service. Used alone, it produces a full listing of APARs against the Network File System.

1. The component identification number for the Network File System is 5695DF121.
2. Go to "Release level keyword".

Release level keyword

Using this keyword to identify the release level of the Network File System is optional in the IBMLink/Service search argument. However, it is required when you submit an APAR.

1. List the System Modification Program Extended (SMP/E) control data set (CDS).
2. In the name column, find the name of the module that appears to be causing the problem.

3. In the entry for the module, find the function modification identifier (FMID) field.
4. The release level keyword for z/OS NFS is R1TS for FMID HDZ11TS or R1TC for FMID HDZ11TC.
5. See “Type-of-failure keyword”.

Type-of-failure keyword

To identify the type of failure that occurred, select the one keyword from Table 36 which best describes the problem. Then follow the specific instructions for that keyword. If you are not certain which of two keywords to use for the type of failure, use the one that appears first in the list.

Table 36. Summary of type-of-failure keywords

Keyword	Type of Failure	Page
ABENDxxx	Network File System ends abnormally because of a system-detected error	93
MSGGFShnnnt	Error indicated by a system message	93
WAIT	The program does not seem to be doing anything	93
LOOP	The program is doing something repetitively	94
INCORROUT	Output from the program is incorrect or missing	94
DOC SCnnnnnnnn	Documentation is incorrect or incomplete	94
PERFM	Performance of the program is degraded	95

ABENDxxx

Use this procedure when the Network File System ends abnormally.

In this step, you build a type-of-failure keyword:

1. Replace the *xxx* part of the **abendxxx** keyword with the abend code from either the SYSLOG message or the abend dump. For example, if the abend code is 0C4, the ABENDxxx keyword is ABEND0C4.
2. To continue, see “Service level keyword” on page 95 to identify the service level of the Network File System.

MSGGFShnnnt

Use this procedure for any of these conditions:

- Network File System message indicates an unexpected error detected.
- Message is not issued under conditions that should cause it to be issued.
- Message is issued under conditions that should not cause it to be issued.

In this step, you build a type-of-failure keyword:

1. For the *hnnnt* portion of the MSGGFShnnnt keyword replace the *h* with an **A** if the message prefix was GFSA or with a **C** if the message prefix was GFSC. Replace the *nnn* with the message number and replace the *t* with the type code. For example, if the message is GFSA865I, the MSGGFShnnnt keyword is MSGGFSA865I.
2. To continue, see “Service level keyword” on page 95 to identify the service level of the Network File System.

WAIT

Use this procedure when the Network File System suspends activity while waiting for some condition to be satisfied without issuing a message to tell why it is waiting.

In this step, you build a type-of-failure keyword:

1. If a Network File System task was in a WAIT, use the **wait** keyword.

5695DF121 Rccc WAIT

2. To continue, see “Service level keyword” on page 95 to identify the service level of the Network File System.

LOOP

Use this procedure when some part of the program repeats endlessly.

In this step, you build a type-of-failure keyword:

1. If a Network File System task was in a LOOP, use the **loop** keyword.
2. If a message repeats endlessly, use the MSGGFShnnnt keyword at the same time with the LOOP keyword. For the *hnnnt* portion of the MSGGFShnnnt keyword replace the *h* with an A if the message prefix was GFSA or with a C if the message prefix was GFSC. Replace the *nnn* with the message number and replace the *t* with the type code.

5695DF121 Rccc LOOP MSGGFShnnnt

3. To continue, see “Service level keyword” on page 95 to identify the service level of the Network File System.

INCORROUT

Do not use **incorROUT** if another keyword applies. Use this procedure only for these conditions:

- Output was expected but not received (missing).
- Output is different from expected (incorrect).

In this step, you build a type-of-failure keyword:

1. Use the **incorROUT** keyword.
2. If the output is in the form of an incorrect message, also use the type-of-failure keyword MSGGFShnnnt. For the *hnnnt* portion of the MSGGFShnnnt keyword replace the *h* with an A if the message prefix was GFSA or with a C if the message prefix was GFSC. Replace the *nnn* with the message number and replace the *t* with the type code.

5695DF121 Rccc INCORROUT MSGGFShnnnt

3. To continue, see “Service level keyword” on page 95 to identify the service level of the Network File System.

DOC SCnnnnnnnn

Use this procedure when a program problem appears to be caused by incorrect or missing information in the Network File System documentation. If the documentation error is minor (for example, incorrect punctuation or a misspelled word), you do not need to build a keyword string to describe it. Instead, submit a Reader’s Comment Form from the back of the book.

If the error is serious, and of general concern to other users, continue with this procedure.

In this step, you build a type-of-failure keyword, which includes a document order number. No other keywords are needed:

1. Place the order number of the document after the DOC keyword, omitting the hyphens. If the suffix is one digit, precede it with a zero. For example, the keyword for a document with order number SC26-7029-01 would be DOC SC26702901.
2. Locate the page in the document at which the error occurs, and prepare a description of the problem. If you submit an APAR, you must include this information in the error description.
3. To continue, see “Service level keyword” to identify the service level of the Network File System.

PERFM

Performance problems can be related to system tuning. Use this procedure when the performance problem cannot be corrected by system tuning and performance is below stated expectations.

In this step, you build a type-of-failure keyword:

1. Use the *perfm* keyword.

```
5695DF121 Rccc PERFM
```

2. To continue, see “Service level keyword” to identify the service level of the Network File System.

Service level keyword

Use this keyword to identify the service level of the module that failed. The service level is defined as the most current fix applied to the module.

1. Locate the service level either in the SDUMP header file or in the copyright information at the beginning of the failure-related CSECT.
2. Specify the service level keyword.

For example, if the service level of the failure related module is UW12345, you would specify UW12345 as the service level keyword.

```
5695DF121 Rccc MSGGFA865I UW12345
```

You now have all the necessary information for an effective search of known problems in IBMLink/Service. Refer to “Contacting the IBM support center” on page 99 or “Searching the IBM database for APARs and PTFs” on page 98.

Log data sets

The Network File System stores the messages in the z/OS NFS client log data sets specified in the NFSCMSG1 and NFSCMSG2 DD statements of the client’s startup procedure.

NFSCMSG1 is associated with the primary log, while NFSCMSG2 is associated with the secondary log. When the client is started, the primary log is used first.

When the primary log is full, the logging is automatically switched to the secondary log and a console message is displayed. The last log buffer is lost when the switch takes place. When the secondary log is also full, the logging is switched back to the primary log and the original primary log content is overwritten.

The Network File System stores the messages in the z/OS NFS server log data sets specified in the NFSLOG1 and NFSLOG2 DD statements of the server's startup procedure.

NFSLOG1 is associated with the primary log, while NFSLOG2 is associated with the secondary log. When the server is started, the primary log is used first. When the primary log is full, the logging is automatically switched to the secondary log and a console message is displayed. The last log buffer is lost when the switch takes place. When the secondary log is also full, the logging is switched back to the primary log and the original primary log content is overwritten.

The number of log records depends on the log level setting and the number of transactions that the server handles. Adjust the space allocation of your z/OS NFS server log data sets according to your installation experience to avoid frequent log switching.

Setting up the z/OS NFS client log data sets

Here is an example of setting up the z/OS NFS client log data sets:

```
//NFSCMSG1 DD DISP=SHR,DSN=MVSNFS.CLIENT.LOG1
//NFSCMSG2 DD DISP=SHR,DSN=MVSNFS.CLIENT.LOG2
```

File attributes of the z/OS NFS client log data sets can be set up like this:

```
Organization      PS
Record Format      VB
Record Length     137
Block Size        6144
```

Reading the message(s) in the z/OS NFS client log data set can help you identify the user correctable error or the problem error which you can report to IBM. See "Client messages" on page 128 for an explanation of the format of the messages that appear in the z/OS NFS client log data sets.

This example shows how to identify a sample problem by reading the z/OS NFS client log data set:

```
13:00:19 GFSC098I (D) CPARS 03 P_MOUNT :
      POS=32 START(7F602A78)
13:00:19 GFSC098I (D) CPARS 02 P_PARM :
      TOKEN.POS(36)TYPE(80)VALUE(0) START(7F602A98)CUR(7F602A9D)
13:00:19 GFSC313I (I) CPARS 04 P_CHECK :
      RETRANS OPTION WILL BE IGNORED AS HARD OPTION IS ON.
13:00:19 GFSC098I (D) CVMNT 74 MI_BLDNR:
      RMN(7F2BAE30:7F172530:7F1728AC) CNT=7
:
:
```

The message GFSC313I in the example is an informational message (indicated by the message level I).

```
13:00:19 GFSC313I (I) CPARS 04 P_CHECK :  
RETRANS OPTION WILL BE IGNORED AS HARD OPTION IS ON.
```

The message text indicates that the keyword, *retrans*, will be ignored because *hard* has been set on. The system programmer should check the mount parameters to make sure that *hard* should be on.

Setting up the z/OS NFS server log data set

Here is an example of setting up the z/OS NFS server log data sets:

```
//NFSLOG1 DD DISP=SHR,DSN=MVSNFS.PROCESS.LOG1  
//NFSLOG2 DD DISP=SHR,DSN=MVSNFS.PROCESS.LOG2
```

File attributes of the z/OS NFS server log data sets can be set up like this:

```
Organization PS  
Record Format VB  
Record Length 137  
Block Size 6144
```

See “Chapter 5. Operating the Network File System” on page 65 and “Log=msglevel operand” on page 73 to find out how to set the level of messages to be stored in this data set. The default message level is “info” which means that all error, attention, and informational messages are collected and stored in the z/OS NFS server log data set. Reading the message(s) in the z/OS NFS server log data set can help you identify the user correctable error or the problem error which you can report to IBM. See “Server messages” on page 103 for an explanation of the format of the messages that appear in the z/OS NFS server log data sets.

This example shows how to identify a sample problem by reading the z/OS NFS server log data set. This example shows a z/OS NFS server log data set with the message level set as “info” (informational):

```
20:25:16 GFSA305I (I) GFSAMAIN ACOPR 01 OPRPARSE: RECEIVED  
COMMAND: EXPORTFS , LENGTH = 9  
20:25:16 GFSA865I (E) GFSAMAIN ANEXP 03 DOOPTION: EXPORTS:  
UNEXPECTED OPTION (PO)--EXPORTS FILE UNUSABLE.  
20:25:16 GFSA866I (E) GFSAMAIN ANEXP 07 EXPORTAL: EXPORTS:  
DIRECTORY (TCP) WAS NOT EXPORTED.  
20:25:16 GFSA925I (E) GFSAMAIN AXOPE 02 OPR_EXPO: ERROR WAS  
DETECTED IN THE EXPORTS FILE. EXPORT LIST NOT REBUILT  
20:25:34 GFSA305I (I) GFSAMAIN ACOPR 01 OPRPARSE: RECEIVED  
COMMAND: FLUSHLOG , LENGTH = 9  
  
:
```

The message GFSA865I in the example is an error message (indicated by the message level E).

```
20:25:16 GFSA865I (E) GFSAMAIN ANEXP 03 DOPTION: EXPORTS:
UNEXPECTED OPTION (PO)--EXPORTS FILE UNUSABLE.
```

The message text indicates that the option “PO” specified in the DIRECTORY statement in the exports data set is not a valid option. You could correct this error by changing “PO” to “RO” and restarting the server.

Flushing the z/OS NFS server log

The messages are buffered in storage. To flush the messages to the z/OS NFS server log data set, issue the **flushlog** operand.

flushlog is an operand of the **modify** operator command which enables you to flush the z/OS NFS server log to disk. For example, you could enter the command:

```
f mvsnfs,flushlog
```

This command enables a TSO user to browse all the log records that have been written by the Network File System.

Setting up a dump data set for abends

Normally, the Network File System ESTAE issues a SVC dump when failure occurs. However, if ESTAE is not able to do this, MVS takes over and issues the appropriate dump you coded in your DD statement. This is an example of setting up a DUMP data set:

```
//SYSMDUMP DD DISP=SHR,DSN=MVSNFS.SYSMDUMP
```

File attributes of this DUMP data set should be set up like this:

```
Organization    PS
Record Format    FB
Record Length   4160
Block Size      4160
```

Searching the IBM database for APARs and PTFs

If your installation has access to the interactive online database program, IBMLink/ServiceLink, you can use IBMLink/ServiceLink to perform these tasks:

1. Search and browse for an existing APAR that is similar to your problem. Use the full set of keywords that is developed from the diagnostic procedures. Use only the keywords that are described in this book. Make sure that keywords are spelled exactly as they are described in this book.
2. If an APAR exists, search to see if a program temporary fix (PTF) is available.
3. If a PTF is available, order the PTF.
4. If an APAR does not exist, you can create an Electronic Technical Response (ETR) problem report to receive assistance from a z/OS NFS service representative. See “Contacting the IBM support center” on page 99 for the type of information you will need.

Contacting the IBM support center

If your installation does not have access to IBMLink/ServiceLink, z/OS NFS also provides telephone support. Within the U.S. and Puerto Rico, call the following number and request assistance for the z/OS NFS feature by specifying the program number 5695DF121 and release level keyword:

IBM Support Center
1-800-237-5511
Monday through Friday, 8 a.m.-5 p.m. (excluding national holidays)

Outside of the U.S.A. and Puerto Rico, contact your local IBM representative.

When contacting IBM, be prepared to supply the following information:

- Your customer number
- Release level
- Current service level (from the APAR list)
- Keyword set or sets used to search in IBMLink/Service

You will be asked to describe the Network File System server and client machine environment. The IBM support representative will request relevant information which could include:

- A minimum set of input commands on the client machine or MVS operator console that reproduces the error.
- A copy of the minimum output from the client machine or MVS operator console necessary to illustrate the failure.
- A copy of the z/OS NFS server log data set, the z/OS NFS client log data set, or both log data sets created by the input commands provided to recreate the error.
- Storage dump (if for abend).
- Linkedit map (if for abend).
- Other supporting material, such as trace file printout from a network analyzer.
- For DOC SC26-7419 (*z/OS Network File System User's Guide*) and DOC SC26-7417 (SC26-7417) failures, include the revision number and page(s) containing the error, and a description of the problem it caused.
- A copy of the attributes data set.
- A copy of the exports data set.
- A copy of the Network File System startup procedure.

Submitting Documentation on Tape: If the IBM service representative requests you submit documentation on tape, please write it to a standard label tape and include a hard copy of the DCB information for each data set along with the JCL used to create the tape.

Diagnostic aids

A description of first failure data capture, including SVC dump characteristics, dump contents, and errors and messages, is provided with the NFS as a major diagnostic aid.

First failure data capture

Network File System RAS characteristics are improved by the capture of diagnostic service data. Error records are written to SYS1.LOGREC and dumps are requested to SYS1.DUMPnn (these are in addition to the existing server trace).

Component-specific information is provided in the SYS1.LOGREC entry and in the dump for the generation of RETAIN[®] search symptom strings.

Symptom data

Table 37 lists the component-specific symptom data placed in the System Diagnostic Work Area (SDWA).

Table 37. NFS symptom data

SDWA Field	Meaning	RETAIN Key
SDWAMODN	Active load module name	RIDS/
SDWAC SCT	Active CSECT name	RIDS/
SDWAMDAT	Active CSECT assembly date	VALU/C
SDWAMVRS	Active CSECT service level	VALU/C
SDWAREXN	Recovery routine module name	RIDS/
SDWARRL	Recovery routine label name	FLDS/
SDWACID	Component identifier	PIDS/
SDWACIDB	Base component identifier	PIDS/
SDWASC	Active server function name	RIDS/

SVC dump

The Network File System SVC dumps have these characteristics:

Dump title: The dump title contains the component name, component identifier, release level, abend code, reason code, and the name of the ESTAE module requesting the dump. If available, the name of the failing module and the offset within the module are included.

Dump content: Table 38 shows the dump options and areas of storage that are included in the dump request:

Table 38. Dump content and storage areas

Dump Options	Storage Areas
SUMDUMP	Suspend summary dump
RGN	Server private area storage; programs and subpools
TRT	MVS trace table
GRSQ	GRS ENQ control blocks
IO	I/O data areas
ALLPSA	All Prefixed Storage Areas
DFA	Data Facility Area
DFVT	Data Facility Vector Table
NFSSVT	Network File System Vector Table

Eye-catchers: Each CSECT within each server load module is identified by the CSECT name, the compile date, and the FMID of “JDZ1170” or APAR level. Each function within a CSECT is identified by its variable length name.

Dump suppression

MVS Dump Analysis and Elimination (DAE) is supported by providing sufficient information for DAE to uniquely identify the dump and by setting the VRADAE key in the Variable Recording Area (VRA) of the SDWA.

Data capture suppression

SYS1.LOGREC entries and SVC dumps are not requested for these ABEND codes:

X'0F3'
X'806'
X'A03'
X'x13'
X'x22' (except X'122')
X'x37'
X'x3E'

Invocation

A server ESTAE instance is entered whenever any server task abends. It is the ESTAE's responsibility to ensure that adequate and correct diagnostics are captured.

Errors and messages

Table 39 illustrates the diagnostic errors and messages GFSA470I and GFSA471I.

Message GFSA470I is written if the SVC dump request fails. The message contains the error reason code from the SDUMP service.

Secondary ESTAE routines detect failures during the execution of the primary server ESTAE. If ESTAE processing is unable to complete, message GFSA471I is issued, and the server task is allowed to stop. The message contains the last abend code detected by the secondary ESTAE routines.

Table 39. Diagnostic errors and messages

Message	Explanation
GFSA470I	NETWORK FILE SYSTEM SVC DUMP REQUEST FAILED. REASON = hh
GFSA471I	NETWORK FILE SYSTEM ESTAE EXIT UNABLE TO COMPLETE PROCESSING. ABEND = X'xxx'

Environmental checklist

This environmental checklist covers useful information that is recommended prior to the initializing the NFS server.

- **Dispatching Priority** - Ensure that z/OS NFS has lower dispatching priority than TCP/IP. Also ensure that TCP/IP has a lower dispatching priority than VTAM®.
- **TCP/IP** - Ensure that the MTU (Packet Size) is set to the lowest MTU when in a heterogeneous network. For example, if the network is comprised of:

Ethernet 802.3 (MTU=1492)
Ethernet Version 2 IEEE (MTU=1500)
token ring (MTU=2000)
FDDI (MTU=4000)
CTC (MTU=65527)
CLAW (MTU=4096)

In this example, the lowest MTU is set to 1492 to reduce packet fragmentation. The MTU setting is defined in the TCP/IP profile.

- **NFS** - Verify that NFS is fully initialized.
 - Ensure that **portmapper** is up.
 - Ensure that NFS has obtained port 2049 for NFS program such as *rpinfo -p <hostname>*.
 - Ensure the NFS has z/OS UNIX SEGMENT with UID=0 defined. the NFS server also needs OPERATIONS RACF authority).
 - For more information, see “Configuring the z/OS NFS server” on page 23 and “Configuring the z/OS NFS client” on page 20.

Useful Utilities: The following utilities are available on client machines help diagnosing simple network connection:

- *rpinfo -p <hostname>* to determine if **portmapper** (port 111), NFS (port 2049) are initialized with the appropriate port.
- *ping* to confirm that there is a live TCP/IP connection between client/sever machines.
- *tracroute <hostname>* to determine how packets are being routed from client to server.
- *iptrace* (AIX) or *snoop* (Solaris) are useful packet tracing utilities used to debug inbound and outbound packets between client and server. For example, during a mount request, using *iptrace* or *snoop* will show whether the client transmitted the mount request, the server has received the request, or the server is still processing the request.

Appendix A. Network File System messages

This appendix lists messages from the NFS server, the NFS client, and the client operating system.

Table 40 contains directive information about this chapter's contents:

Table 40. Messages - client operating system, NFS server, and NFS client

Section	Page
Server messages	103
Client messages	128
Messages from the client platform (AIX)	145

Server messages

This is a listing of the messages generated by the NFS server. For each message, explanations and recommended actions are given where applicable. Data is substituted for any part of a message shown here in *italics*.

Messages that appear in the NFS server log data set in the same format as this example: **19:25:14 GFSA348I (I) GFSAMAIN ANMAI 02 NFS_INIT: OS/390 VERSION 1 NETWORK FILE SYSTEM SERVER (HDZ11TS) STARTED.**

Table 41 shows the NFS server log data set message format:

Table 41. NFS server log data set message format

19:25:14	The time stamp (hours:minutes:seconds).
GFSA	Component identifier for the NFS server.
348	A unique message number.
I	Message type: A Action; the user must perform a specific action. E Eventual action; the user must perform an action when time is available. I Informational; no user action is required.
(I)	The message level: E (error), W (attention), or I (informational). The system programmer can use the message level to determine which type of messages are shown by specifying log=error, log=warn, and log=info levels.
GFSAMAIN ANMAI 02 NFS_INIT:	Programming support information.
OS/390 VERSION 1 NETWORK FILE SYSTEM SERVER (HDZ11TS) STARTED.	Message text.

The messages are listed in numerical order (the time stamp, message level, and programming support information are not shown).

Notes:

1. *h_digits* denotes hexadecimal digits, and *d_digits* denotes decimal digits. *text* represents variable text (such as a data set name).
2. Messages GFS300I through GFS319I are intended for IBM support personnel when they are performing diagnosis.
3. For messages written to the console, the name of the start procedure is substituted for <PROCNAME>.

**GFS320I(<PROCNAME>) NETWORK FILE
SYSTEM SERVER INITIALIZATION
FAILED: *text***

Explanation: *text* can be:

- VIRTUAL STORAGE IS UNAVAILABLE
- MAPPING SIDE FILE NOT FOUND
- MAPPING SIDE FILE HAS INVALID SYNTAX OR FORMAT
- ERROR OPENING/READING MAPPING SIDE FILE
- SIDE FILE SPECIFIED BUT MAPPING IS DIS-ALLOWED BY INSTALLATION
- TASK IS NOT APF AUTHORIZED
- DFP LEVEL MUST BE DFSMS 1.2 OR HIGHER
- SERVER ALREADY STARTED

System Action: The NFS startup ends.

Operator Response: Notify the system programmer.

System Programmer Response: The system programmer response is:

- If it is a virtual storage problem, increase the region size.
- If an APF-authorization problem, APF authorize all libraries in the STEPLIB DD statement.
- If mapping side file is not found, make sure the name specified in the attribute data set is correct and the file exists.
- If mapping side file has invalid syntax, check the part GFSAPMAP in SYS1.NFSSAMP library for mapping side file rules.
- If error occurs during opening of the side file, check to make sure that the side file is not migrated and it is readable.
- If s_fmax=0 then the side file can not be specified in the attribute data set.
- If a back-level release of DFP, restart NFS server after installing DFSMSd_fp™ 1.2 or a later release.
- If the same start procedure is used with the same TCP/IP stack name, use a different name for either the start procedure or the TCP/IP stack name.

**GFS321I(<PROCNAME>) NETWORK FILE
SYSTEM SERVER INITIALIZATION
FAILED: OS/390 UNIX ADDRESS
SPACE HAS NOT BEEN STARTED.**

Explanation: The NFS was not able to establish successful communication with the UNIX address space.

System Action: The NFS ends.

Operator Response: Before starting the NFS, both the UNIX and the TCP/IP address spaces must have successfully been started.

**GFS322I(<PROCNAME>) OPENEDITION V_REG
FAILED: RV=1, RC=*h_digit1*,
RSN=*h_digit2*.**

Explanation: The NFS failed to register.

System Action: The NFS ends.

Operator Response: Contact the system programmer.

System Programmer Response: *h_digit1*, *h_digit2* are the return code and reason code from OpenEdition V_REG callable service. See *z/OS UNIX System Services File System Interface Reference* for more information about return code and reason codes.

**GFS323I OS/390 NETWORK FILE SYSTEM LOCK
MANAGER (HDZ11TS) WILL NOT
START BECAUSE OMVS(UID(0)) IS
NOT DEFINED.**

Explanation: The Network Lock Manager (NLM) initialization fails because the NLM startup procedure is not defined as OMVS(UID(0)).

System Action: The NLM terminates.

User Response: Define NLM startup procedure as OMVS(UID(0)).

**GFS324I OS/390 NETWORK FILE SYSTEM
STATUS MONITOR (HDZ11TS) WILL
NOT START BECAUSE OMVS(UID(0))
IS NOT DEFINED.**

Explanation: The Network Status Monitor (NSM) initialization fails because the NSM startup procedure is not defined as OMVS(UID(0)).

System Action: The NSM terminates.

User Response: Define NSM startup procedure as OMVS(UID(0)).

GFS325I REQUESTED MEMORY NOT AVAILABLE.

Explanation: An operation to allocate virtual memory was tried, but was unsuccessful. If this condition persists, the cause might be:

- The value specified on the **region** parameter is too small.
- The value specified on the **bufhigh** attribute is too large.

System Action: The request is stopped. NFS processing continues.

GFS328I(<PROCNAME>) NO SWAP REQUEST FOR NETWORK FILE SYSTEM SERVER FAILED.

Explanation: The MVS NO SWAP request for NFS has failed.

System Action: The Network File System ends.

Operator Response: Contact the system programmer.

System Programmer Response: Try to start the NFS server again. If the failure appears to be a Network File System error, contact IBM service. Have available a symptom string and a copy of the MVS console log.

GFS329I(<PROCNAME>) SERVER SHUTDOWN IN PROGRESS.

Explanation: Shutdown procedures have started.

System Action: NFS shutdown continues.

GFS330I(<PROCNAME>) SERVER SHUTDOWN COMPLETE.

Explanation: Network File System and its associated subtasks have ended.

System Action: The Network File System stops.

GFS331E(<PROCNAME>) RECALL FAILED FOR MIGRATED DATA SET *text*.

Explanation: DFSMSHsm was unable to recall a data set, *text*, because the data movement program DFSMSdss™ detected, during restore, that the migrated data set had internal errors.

System Action: The NFS server processing continues.

Operator Response: Notify system programmer for recovery actions.

System Programmer Response: The data set *text*

had an internal error when migrated, and cannot be recalled. See DFSMSHsm Message ARC0075E for appropriate recovery actions.

GFS333I(<PROCNAME>) OS/390 *version* NETWORK FILE SYSTEM SERVER *fmid* IS NOT STARTED BY A START COMMAND.

Explanation: The system programmer tried to start the Network File System with a command other than **start**.

In the message text:

version The version number, for example: 1.1.0 or VERSION 1.

fmid The NFS server FMID started, for example: null or (HDZ11TS).

System Action: The Network File System stops.

System Programmer Response: Start the Network File System by issuing a **start** command.

GFS334I(<PROCNAME>) MOUNT HANDLE DATABASE CANNOT BE READ.

Explanation: The Network File System has tried to read the mount handle database but was unable to do so. This could occur because the mount handle database couldn't be opened or because it contained a record whose length or contents were incorrect.

System Action: System processing continues but in a degraded mode. Some or all the directories have not been remounted.

Operator Response: Contact the system programmer.

System Programmer Response: If this is the first startup after installation of a new release, the mount handle data sets must be cleared and the server restarted, because the format of the mount handle database is not recognizable by the new release. Clients have to reboot or unmount any previously mounted directories.

GFS335E(<PROCNAME>) MOUNT HANDLE DATA SET CANNOT BE WRITTEN, EXPECTING LEN *d_digits1* REAL LEN *d_digits2*, VSAM R15(DEC) *d_digits3* REASON CODE(DEC) *d_digits4* LAST OP(DEC) *d_digits5*.

Explanation: The Network File System has tried to write a mount record to the mount handle database but was unable to do so. The length of the mount record is *d_digits1* bytes, but only *d_digits2* bytes were written. The failing information in writing the VSAM KSDS mount handle database is the decimal return code *d_digits3*, the decimal error code or reason code *d_digits4*, and the code for the last operation *d_digits5*.

System Action: System processing continues but in a degraded mode. Any further mounts or unmounts are likely to fail in writing the record in mount handle data set.

Operator Response: Contact the system programmer.

System Programmer Response: The mount handle data sets have probably become unusable and either need to be cleared or restored to some previous level, and then the server has to be restarted. The information for the mount point was not saved in the mount handle data set. Clients might have to unmount and mount these mounted directories when the server is restarted.

GFSA335I MOUNT HANDLE DATA SET CANNOT BE WRITTEN, EXPECTING LEN *d_digits1* REAL LEN *d_digits2*, VSAM R15(DEC) *d_digits3* REASON CODE(DEC) *d_digits4* LAST OP(DEC) *d_digits5*.

Explanation: The Network File System has tried to write a mount record to the mount handle database but was unable to do so. The length of the mount record is *d_digits1* bytes, but only *d_digits2* bytes were written. The failing information in writing the VSAM KSDS mount handle database is the decimal return code *d_digits3*, the decimal error code or reason code *d_digits4*, and the code for the last operation *d_digits5*.

System Action: System processing continues but in a degraded mode. Any further mounts or unmounts are likely to fail in writing the record in mount handle data set.

Operator Response: Contact the system programmer.

System Programmer Response: The mount handle data sets have probably become unusable and either need to be cleared or restored to some previous level, and then the server has to be restarted. Clients might have to unmount and mount any previously mounted directories.

GFSA336E(<PROCNAME>) MOUNT HANDLE DATA SET CANNOT BE OPENED, VSAM R15(DEC) *d_digits1* REASON CODE(DEC) *d_digits2* LAST OP(DEC) *d_digits3*.

Operator Response: Contact the system programmer.

System Programmer Response: The mount handle data sets probably created with incorrect attributes.

System Action: The NFS stops.

Explanation: During resource timeout, the NFS had tried to open the mount handle data set for writing but unable to do so. The failing information in fopen the virtual storage access method (VSAM) key-sequenced data set (KSDS) mount handle data set is the decimal return code *d_digits1*, the decimal error code or reason

code *d_digits2*, and the code for the last operation *d_digits3*.

GFSA336I OS/390 NETWORK FILE SYSTEM LOCK MANAGER SHUTDOWN COMPLETE

Explanation: The NFS Network Lock Manager (NLM) shutdown completed successfully.

Module: NFS Network Lock Manager (NLM).

System Action: NFS Network Lock Manager (NLM) ended.

GFSA337I THE FILE SIZE IS TOO LARGE. FN=*text*, OFFSET=*h_digit1*, LEN=*h_digit2*.

Explanation: The Network File System client user tried to read or write the file *text* beyond the 4G.

System Action: The request is stopped. An error, NFSERR_FBIG(27), is returned to the client. NFS processing continues.

User Response: Check the file size used. Split into multiple files and try again.

GFSA338I OS/390 NETWORK FILE SYSTEM STATUS MONITOR SHUTDOWN COMPLETE

Explanation: The NFS Network Status Monitor (NSM) shutdown completed successfully.

Module: NFS Network Status Monitor (NSM).

System Action: NFS Network Status Monitor (NSM) ended.

GFSA344I OS/390 NETWORK FILE SYSTEM LOCK MANAGER (*fmid*) STARTED

Explanation: The NFS Network Lock Manager (NLM) is initialized successfully and is started.

Module: The NFS Network Lock Manager (NLM).

System Action: The NFS Network Lock Manager (NLM) startup ends and continues processing.

GFSA345I OS/390 NETWORK FILE SYSTEM STATUS MONITOR (*fmid*) STARTED

Explanation: The NFS Network Status Manager (NSM) is initialized successfully.

Module: The NFS Network Status Manager (NSM).

System Action: The NFS Network Status Manager (NSM) startup ends and continues processing.

GFSA346I *time_stamp*.

Explanation: Displays the current time stamp. This message is issued when the NFSLOG switches.

GFSA347I **ERROR RETURNED TO CLIENT: RC =**
d_digits <text>.

Explanation: The error code *d_digits* was returned to the client. *text* is the meaning of the error code.

System Action: Network File System processing continues.

GFSA348I(<PROCNAME>) OS/390 *version*
NETWORK FILE SYSTEM SERVER *fmid*
STARTED.

Explanation: The Network File System is initialized and ready to accept **modify** commands from the operator console.

In the message text:

version The version number, for example: 1.1.0 or
VERSION 1.

fmid The NFS server FMID started, for example:
null or (HDZ11TS).

System Action: Network File System continues processing.

GFSA349I **UNEXPECTED ERROR DETECTED:**
d_digits text.

Explanation: The Network File System has encountered a condition that indicates continued processing might produce undesirable results. *text* is additional debugging information for the programming support personnel.

System Action: The Network File System either shuts down or stops the request and continue processing, depending on where the error was detected.

System Programmer Response: Contact your programming support personnel.

GFSA360I *text*.

Explanation: This message displays memory management statistics, *text*.

System Action: The Network File System continues processing.

GFSA361I **NETWORK FILE SYSTEM IS SHORT
ON STORAGE.**

Explanation: This message is displayed to the operator console when a shortage of virtual storage is detected.

System Action: The Network File System continues processing. The storage constraint might be relieved when some storage is freed up later on.

Operator Response: If this message is displayed repeatedly within a short period of time, stop or cancel the Network File System and notify the system programmer.

System Programmer Response: Do one or both of the following before restarting the Network File System:

1. Increase the region size for the step or started task.
2. Decrease the value specified for the *bufhigh* attribute of the attributes data set.

**GFSA362I(<PROCNAME>) REGION SIZE WILL NOT
ACCOMMODATE BUFHIGH AND
LOGICAL CACHE SPECIFICATIONS.**

Explanation: The values of *bufhigh* and *logicalcache* specified in the attributes data set are incorrect.

System Action: The Network File System startup stops.

System Programmer Response: Either increase the region size of the step, or reduce the total of the *bufhigh* and *logicalcache* attribute values.

GFSA400I **INVALID RECFM SPECIFICATION (*text*).**

Explanation: *text* is the incorrect record format specified in the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

**GFSA401I(<PROCNAME>) CANNOT OPEN THE
ATTRIBUTE DATA SET.**

Explanation: The server was unable to open the attributes data set defined in the JCL for DDNAME NFSATTR. The DD statement might be missing or the data set name might be incorrect.

System Action: The Network File System stops.

Operator Response: Notify your system programmer.

System Programmer Response: Correct the JCL for DDNAME NFSATTR.

GFSA402I(<PROCNAME>) READ FAILED FOR THE ATTRIBUTES DATA SET.

Explanation: An error occurred while the Network File System was processing the attribute data set. This message follows other messages that describe the error in greater detail. The attributes data set is defined in the JCL for DDNAME NFSATTR.

System Action: The Network File System stops.

Operator Response: Notify your system programmer.

System Programmer Response: Correct the attributes data set. **Note:** When the NFS attribute data set is created, the *num off* option in ISPF should also be used.

The sequence number is not allowed in the NFS attributes data set.

GFSA403I(<PROCNAME>) PARSE FAILED IN LINE *d_digits text*.

Explanation: The parsing of line number *d_digits* in the attribute data set failed. *text* is the actual line from the attributes data set that contains the failure. This message follows other messages that describe the error in greater detail.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA404I UNEXPECTED END OF STRING ON END OF PARSE IN LINE *d_digits*.

Explanation: A comma is missing between attributes on line number *d_digits* of the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA405I PARSE FAILED FOR ATTRIBUTE FIELD - ILLEGAL KEYWORD IN LINE *d_digits*.

Explanation: The keyword specified in line number *d_digits* of the attribute data set is not a valid attribute keyword.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA406I MISSING LEFT PARENTHESIS IN LINE *d_digits*.

Explanation: An attribute specified on line number *d_digits* of the attributes data set is missing a left parenthesis.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA407I MISSING RIGHT PARENTHESIS IN LINE *d_digits*.

Explanation: An attribute specified on line number *d_digits* of the attributes data set is missing a right parenthesis.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA408I PARSE FAILED ON NUMBER FIELD IN LINE *d_digits*.

Explanation: An attribute with a negative number was specified on line number *d_digits* of the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA409I INVALID DSORG SPECIFICATION IN LINE *d_digits*.

Explanation: The data set organization specified in the *dsorg* attribute on line number *d_digits* of the attributes data set is incorrect or is not supported by the Network File System.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA411I INVALID VOL SPECIFICATION IN LINE *d_digits*.

Explanation: The volume specified in the *vol* (or volume) attribute on line number *d_digits* of the attributes data set is incorrect.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA412I INVALID UNIT SPECIFICATION IN LINE *d_digits*.

Explanation: The unit specified in the unit attribute on line number *d_digits* of the attributes data set is incorrect.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA413I THE ATTRIBUTE VALUE *d_digits1* IS NOT IN THE RANGE OF *d_digits2* TO *d_digits3*.

Explanation: The value, *d_digits1*, specified in one of the attributes of the attributes data set must be between the minimum value, *d_digits2*, and the maximum value, *d_digits3*, for this attribute. Message GFSA403I follows this message.

System Action: The Network File System stops.

System Programmer Response: See message GFSA403I to determine the attribute in error, then correct the attributes data set.

GFSA414I THE ATTRIBUTE VALUE *d_digits1* EXCEEDS THE MAXIMUM TIMEOUT VALUE OF *d_digits2*.

Explanation: The value, *d_digits1*, specified in one of the attributes of the attributes data set must be less than or equal to *d_digits2*. *d_digits2* is the maximum value allowed for the attribute. Message GFSA403I follows this message.

System Action: Network File System stops.

System Programmer Response: See message GFSA403I to determine the attribute in error, then correct the attributes data set.

GFSA415I THE ATTRIBUTE TIME OUT VALUE *d_digits1* IS LESS THAN THE MINIMUM TIME OUT VALUE *d_digits2*.

Explanation: The value, *d_digits1*, specified in the *attrtimeout* attribute in the attributes data set must be greater than or equal to the value *d_digits2* which is specified in the *mintimeout* attribute of the attribute data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA416I THE READ TIME OUT VALUE *d_digits1* IS LESS THAN THE MINIMUM TIME OUT VALUE *d_digits2*.

Explanation: The value, *d_digits1*, specified in the *readtimeout* attribute of the attributes data set must be greater than or equal to the value *d_digits2* which was specified in the *mintimeout* attribute of the attribute data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA417I THE WRITE TIME OUT VALUE *d_digits1* IS LESS THAN THE MINIMUM TIME OUT VALUE *d_digits2*.

Explanation: The value, *d_digits1*, specified in the *writetimeout* attribute of the attributes data set must be greater than or equal to the value *d_digits2* which was specified in the *mintimeout* attribute of the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA418I THE ATTRIBUTE TIME OUT VALUE *d_digits1* IS GREATER THAN THE MAXIMUM TIME OUT VALUE *d_digits2*.

Explanation: The value, *d_digits1*, specified in the *attrtimeout* attribute of the attributes data set must be less than or equal to the value *d_digits2* which was specified in the *maxtimeout* attribute of the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA419I THE READ TIME OUT VALUE *d_digits1* IS GREATER THAN THE MAXIMUM TIME OUT VALUE *d_digits2*.

Explanation: The value, *d_digits1*, specified in the *readtimeout* attribute of the attributes data set must be less than or equal to the value *d_digits2* which was specified in the *maxtimeout* attribute of the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFSA420I THE WRITE TIME OUT VALUE *d_digits1* IS GREATER THAN THE MAXIMUM TIME OUT VALUE *d_digits2*.

Explanation: The value, *d_digits1*, specified in the *writetimeout* attribute of the attributes data set must be less than or equal to the value *d_digits2* which was specified in the *maxtimeout* attribute of the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFS4211 THE NOATTRTIMEOUT ATTRIBUTE WAS SPECIFIED BUT THE MAXTIMEOUT VALUE WAS SET TO *d_digits*.

Explanation: The *noattrtimeout* attribute means that the data set is not to be deallocated following a **lookup** or **getattr** operation. The *maxtimeout* attribute specifies the maximum time out value allowed for any of the timeout attributes. *d_digits* was specified as the maxtimeout value, in seconds, that the data set is to remain allocated. These attributes are in conflict. The *noattrtimeout* and *maxtimeout* attributes are specified in the attributes data set.

System Action: The Network File System stops.

System Programmer Response: If the *noattrtimeout* attribute is desired, the *nomaxtimeout* attribute must be specified in the attribute data set. Correct the attributes data set.

GFS4221 THE NOREADTIMEOUT ATTRIBUTE WAS SPECIFIED BUT THE MAXTIMEOUT VALUE WAS SET TO *d_digits*.

Explanation: The *noreadtimeout* attribute means that the data set is not to be deallocated following a read operation. The *maxtimeout* attribute specifies the maximum time out value allowed for any of the timeout attributes. *d_digits* was specified as the value of the *maxtimeout* attribute, in seconds, that the data set is to remain allocated. These attributes are in conflict. The *noreadtimeout* and *maxtimeout* attributes are specified in the attributes data set.

System Action: The Network File System stops.

System Programmer Response: If the *noreadtimeout* attribute is desired, the *nomaxtimeout* attribute must be specified in the attribute data set. Correct the attributes data set.

GFS4231 THE NOWRITETIMEOUT ATTRIBUTE WAS SPECIFIED BUT THE MAXTIMEOUT VALUE WAS SET TO *d_digits*.

Explanation: The *nowritetimeout* attribute means that the data set is not to be deallocated following a write operation. The *maxtimeout* attribute specifies the maximum time out value allowed for any of the timeout attributes. *d_digits* was specified as the value of the *maxtimeout* attribute, in seconds, that the data set is to remain allocated. These attributes are in conflict. The *nowritetimeout* and *maxtimeout* attributes are specified in the attributes data set.

System Action: The Network File System stops.

System Programmer Response: If the *nowritetimeout* attribute is desired, the *nomaxtimeout* attribute must be specified in the attribute data set.

Correct the attributes data set.

GFS4241 MINIMUM TIME OUT VALUE, *d_digits1*, IS GREATER THAN THE MAXIMUM TIME OUT VALUE, *d_digits2*.

Explanation: The value, *d_digits1*, specified in the *mintimeout* attribute of the attributes data set is greater than the value, e *d_digits2*, specified in the *maxtimeout* attribute of the attributes data set.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFS4251 INVALID SPECIFICATION RECFM(*text*).

Explanation: One of the characters in *text* specified on the *recfm* attribute of the attributes data set is incorrect.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFS4261 INVALID RECFM(*text*) - MUST SPECIFY U, F, OR V.

Explanation: One of the characters in *text* specified on the *recfm* attribute of the attributes data set must define whether the records are fixed length (F), variable length (V), or undefined (U) format records.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFS4291 INVALID DSNTYPE SPECIFICATION IN LINE *d_digits*.

Explanation: The data set name type specified in the *dsntype* attribute on line number *d_digits* of the attributes data set is incorrect.

System Action: The Network File System stops.

System Programmer Response: Correct the attributes data set.

GFS4301 INVALID SMS_keyword SPECIFICATION IN LINE *d_digits*.

Explanation: The SMS keyword *SMS_keyword* shown is syntactically incorrect on line number *d_digits*. See system-managed storage documentation for DATACLAS, MGMTCLAS, and STORCLAS naming conventions.

System Action: Network File System startup ends if the keyword was specified as a site attribute. If the incorrect SMS keyword was specified by a client as a mount parameter or on a command, the line number is set to zero and an I/O error is returned to the client.

System Programmer Response: Correct the site attributes file, if applicable.

**GFSA431I INVALID OPTION SPECIFICATION OF
text IN LINE *d_digits*.**

Explanation: The option specified in the **text** attribute on line number *d_digits_* of the attributes data set is incorrect.

System Action: Network File System stops.

System Programmer Response: Correct the site attributes file, if applicable.

**GFSA434I(<PROCNAME>) text1(*d_digits1*) IS SET
TO THE DEFAULT VALUE,
text1(*d_digits2*,*d_digits3*,*d_digits4*).**

Explanation: *text1(d_digits1)* is the value from the previous release. This value could not be applicable to the new release. For forward compatibility, this value takes on the default value for the new release.

System Action: Network File System processing continues.

**GFSA435E SUM OF *d_digit1* PLUS *d_digits2* MUST
BE LESS THAN OR EQUAL TO *d_digit3*.**

Explanation: The sum of *d_digit1* plus *d_digit2* is greater than *d_digit3*.

System Action: Network File System processing stops.

System Programmer Response: Correct the values.

**GFSA436I INVALID SIDEFIELD SPECIFICATION IN
LINE *d_digits***

Explanation: A data set name was not specified in the sidefile attribute.

System Action: The NFS server startup ends.

System Programmer Response: Correct the problem and make necessary changes in the attribute data set.

**GFSA437I INVALID PUBLIC SPECIFICATION IN
LINE *d_digits***

Explanation: The parsing of the *public* keyword resulted in error for one of the following reasons:

- *public* keyword specification is syntactically incorrect
- No public pathnames have been specified
- Pathname specified is invalid
- HFS public pathname does not match HFS prefix

System Action: The NFS server startup ends

Operator Response: Correct the problem and make the necessary changes in the attributes data set.

**GFSA438I EXPORT SPANNING PATHNAMES NOT
SUPPORTED**

Explanation: The Export Spanning Pathnames support for a multi-component **lookup** request is not supported.

System Action: The request fails. NFS processing continues.

User Response: Construct a different pathname in which the path is not spanned.

**GFSA439I HFS PUBLIC PATHNAME SPECIFIED
BUT HFS IS NOT ENABLED**

Explanation: An HFS public pathname was specified in the **public** keyword but *nohfs* was also specified which disables HFS processing.

System Action: The NFS server startup ends.

Operator Response: Correct the problem and make the necessary changes in the installation table.

**GFSA440I INVALID SECURITY SPECIFICATION IN
*d_digits***

Explanation: The parsing of the *security* keyword resulted in error for one of the following reasons.

- Missing first parameter
- Invalid first parameter specified

System Action: The NFS server startup ends.

Operator Response: Correct the problem and make necessary changes in the attributes data set.

GFSA450I CREATED TASK(*h_digits*) - *text1* - *text2*.

Explanation: The Network File System is creating the number of tasks requested in the **nfstasks** attribute of the attributes data set. This message is displayed for each task created. *h_digits* is the TCB address. *text1* is the task name. *text2* is the module name.

System Action: Network File System processing continues.

GFSA451I DELETING TASK(*h_digits*) - *text*.

Explanation: The Network File System is deleting a task. This is in response to the **stop** operand of the **modify** command. This message is displayed for each task deleted. *h_digits* is the TCB address. *text* is the module name.

System Action: Network File System shutdown continues.

GFSA452I SUBTASK TERMINATED: *h_digits*.

Explanation: The Network File System is stopping a task. This is in response to the **stop** operand of the **modify** command. This message is displayed for each ended task. *h_digits* is the TCB address.

System Action: Network File System shutdown continues.

**GFSA470I(<PROCNAME>) NETWORK FILE SYSTEM SERVER SVC DUMP REQUEST FAILED.
REASON=*reason_code*.**

Explanation: A request to write an MVS SVC dump failed. *reason_code* is a hexadecimal value indicating the reason MVS was unable to write the dump. See the description of the **sdump** macro in *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU* for the meaning of the reason code.

System Action: Abend processing continues.

Operator Response: If the dump could not be written due to an operational procedure and a dump is necessary to diagnose the failure, correct the procedure.

System Programmer Response: If the failure appears to be due to a Network File System error, contact IBM service. Have available a symptom string and a copy of the MVS console log.

GFSA471I(<PROCNAME>) NETWORK FILE SYSTEM SERVER ESTAE EXIT UNABLE TO COMPLETE PROCESSING. ABEND=*abend_code*.

Explanation: The Network File System ESTAE exit routine abended and is unable to complete processing. *abend_code* is set to the last abend code encountered by a secondary instance of the ESTAE exit routine.

System Action: ESTAE processing is stopped. The Network File System address space might end also.

Operator Response: Restart the Network File System address space. Notify network users of the failure.

System Programmer Response: Contact IBM service. Have available a symptom string, the related SYS1.LOGREC entries, any related SDUMPs, and the Network File System error trace log.

GFSA480I READ/WRITE REQUEST FAILED BECAUSE OF ERRORS ENCOUNTERED IN THE CONVERSION SERVICES, RC=*d_digits1*, RSN=*d_digits2*.

Explanation: The *READ/WRITE* request received an error condition from the conversion services while data

was being translated. See the error codes in *OS/390 Support for Unicode™: Using Conversion Services* for an explanation of the errors encountered.

System Action: The request failed. NFS processing continues.

User Response: See *OS/390 Support for Unicode™: Using Conversion Services* for information about the error encountered and correct the problem if possible.

GFSA481I READ/WRITE REQUEST FAILED BECAUSE TEXT CONVERSION RESULTED IN THE LENGTHS OF INPUT AND OUTPUT STRINGS BEING DIFFERENT.

Explanation: The *READ/WRITE* request failed because the translation of the text string resulted in a different length. Translations that involved length changes (such as SBCS to MBCS) are not currently supported.

System Action: The request failed. NFS processing continues.

User Response: Make the necessary changes so that text string translations do not result in different lengths.

GFSA482I READ/WRITE REQUEST FAILED BECAUSE TRANSLATION OF TEXT IS NOT POSSIBLE.

Explanation: A *READ/WRITE* operation is being requested for a file that is tagged with a valid Coded Character Set Identifier (CCSID) but the user has not specified a *cln_ccsid* to be used for translation and *xlat()* keyword is also specified.

System Action: The request failed. NFS processing continues.

User Response: Either specify a *cln_ccsid* for translation or remove the specification *xlat()* keyword.

GFSA483I WRITE REQUEST FAILED BECAUSE BINARY DATA CANNOT BE WRITTEN TO A FILE WITH PURE TEXT DATA.

Explanation: A *WRITE* request has binary option specified explicitly and the file being written to has pure text data.

System Action: The request failed. NFS processing continues.

User Response: Either remove the binary option or change the file tag that is associated with the file to show that the file has mixed data.

GFSA501I REQUEST HEADER ALLOCATION FAILED.

Explanation: An operation to allocate virtual memory for a request header was tried, but was unsuccessful.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Increase the size of the step region.

GFSA502I REQUEST HEADER DATA BLOCK ALLOCATION FAILED.

Explanation: An operation to allocate virtual memory for a request header data block was tried, but was unsuccessful.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Increase the size of the step region.

GFSA554I(<PROCNAME>) REGISTER FOR PORT *d_digits1* PROGRAM *d_digits2* - VERSION *d_digits3* - FAILED.

Explanation: *d_digits1* is the port number. *d_digits2* is the remote procedure call (RPC) program number. *d_digits3* is the RPC program version number.

System Action: The Network File System stops.

Operator Response: Check your MVS TCP/IP setup or notify your system programmer.

System Programmer Response: Check your MVS TCP/IP setup.

GFSA555I REGISTER FOR PORT *d_digits1* PROGRAM *d_digits2* - VERSION *d_digits3* - SUCCESSFUL.

Explanation: Information only. *d_digits1* is the port number. *d_digits2* is the remote procedure call (RPC) program number. *d_digits3* is the RPC program version number.

System Action: The Network File System continues.

GFSA556I(<PROCNAME>) UNREGISTER PROGRAM *d_digits1* VERSION *d_digits2* - FAILED.

Explanation: *d_digits1* is the remote procedure call (RPC) program number. *d_digits2* is the RPC program version number.

System Action: Network File System stops.

Operator Response: Contact system programmer.

System Programmer Response: This can be caused

by an out-of-memory error. Increase the size of the job and step region.

GFSA557I UNREGISTER PROGRAM *d_digits1* VERSION *d_digits2* - SUCCESSFUL.

Explanation: Information only. *d_digits1* is the remote procedure call (RPC) program number.

System Action: Network File System processing continues.

GFSA558I(<PROCNAME>) UNABLE TO CREATE IPC QUEUE.

Explanation: An operation to allocate virtual memory for an inter-process communication (IPC) queue was tried, but was unsuccessful.

System Action: The Network File System stops.

Operator Response: Notify your system programmer.

System Programmer Response: Increase the size of the step region.

GFSA559I(<PROCNAME>) CANNOT CREATE UDP SERVICE.

Explanation: User datagram protocol (UDP) service transport could not be created, or you started TCP/IP before the UNIX initialization was completed.

System Action: The Network File System stops.

Operator Response: Check your MVS TCP/IP setup or notify your system programmer. Before starting TCP/IP, make sure that the UNIX initialization has completed, and the TCP/IP UNIX connection is established.

System Programmer Response: Check your MVS TCP/IP setup. Also check your UNIX BPXPRMxx parmlib member, specifically the MAXFILEPROC, MAXSOCKETS, INADDRANYPORT, and INADDRANYCOUNT. The INADDRANYPORT and INADDRANYCOUNT must be specified, but its range can not include 2049, in order for the server to initialize. See *z/OS MVS Initialization and Tuning Guide* for more details.

GFSA564I(<PROCNAME>) NETWORK SERVICE IS UNAVAILABLE.

Explanation: If the **Portmapper** or TCP/IP is not running, the Network File System fails to initialize and the startup ends.

System Action: The Network File System startup ends.

System Programmer Response: Ensure that both the **Portmapper** and TCP/IP are operational.

**GFSA565I(<PROCNAME>) OS/390 NFS SERVER
CANNOT OBTAIN NFS PORT 2049.**

Explanation: The Network File System cannot obtain NFS port 2049.

System Action: The Network File System ends.

Operator Response: Check your MVS TCP/IP setup or notify your system programmer.

System Programmer Response: Check your MVS TCP/IP setup.

**GFSA566I(<PROCNAME>) CANNOT ACCEPT NEW
TCP CLIENT CONNECTION —
MAXIMUM NUMBER OF SOCKETS HAS
REACHED**

Explanation: When an NFS TCP client attempts to connect to the NFS server, the server cannot accept the connection because the maximum number of sockets has been reached.

System Action: The connection request fails. NFS processing continues.

Operator Response: Increase the value of **maxsockets** for AF_INET domain in the BPXPRMxx parmlib member so that client TCP connections can be accepted by the NFS server. See *z/OS UNIX System Services Planning* for more information.

**GFSA750I(<PROCNAME>) SMF PROCESSING
ACTIVE FOR USER LOGOUT.**

Explanation: SMF processing is active for user logout records. This message is displayed in response to the **status** operand on the **modify** command.

System Action: NFS server processing continues.

**GFSA751I(<PROCNAME>) SMF PROCESSING
SUSPENDED FOR USER LOGOUT.**

Explanation: System-managed facility (SMF) processing is suspended for user logout records. This message is displayed in response to the **status** operand on the **modify** command, or after receiving a non-zero return code from SMF. See message GFSA754I for SMF return code.

System Action: NFS server processing continues without generating any user or file SMF records.

Operator Response: Resume SMF recording by entering a **modify** command specifying SMF=0N.

**GFSA752I(<PROCNAME>) SMF PROCESSING
ACTIVE FOR FILE TIMEOUT.**

Explanation: System-managed facility (SMF) processing is active for file timeout records. This message is displayed in response to the **status**

operand on the **modify** command.

System Action: NFS server processing continues.

**GFSA753I(<PROCNAME>) SMF PROCESSING
SUSPENDED FOR FILE TIMEOUT.**

Explanation: System-managed facility (SMF) processing is suspended for file timeout records. This message is displayed in response to the status operand on the **modify** command, or after receiving a non-zero return code from SMF. See message GFSA754I for SMF return code.

System Action: NFS server processing continues without generating any user or file SMF records.

Operator Response: Resume SMF recording by entering a **modify** command specifying SMF=0N.

**GFSA754I(<PROCNAME>) UNEXPECTED RETURN
CODE *d_digits* RECEIVED FROM SMF
WHILE WRITING RECORD TYPE 42
SUBTYPE [7|8].**

Explanation: The NFS server received a non-zero return code *d_digits* while processing a file timeout (subtype 7) or user logout (subtype 8) record.

System Action: Network File System processing continues. No more SMF records of the same type and subtype are generated until the Network File System address space is restarted or the SMF=0N operand of the **modify** command is entered.

Operator Response: Notify the system programmer.

System Programmer Response: See *z/OS MVS System Management Facilities (SMF)* for information about the return code. Correct the problem, and have the operator enter a **modify** command specifying SMF=0N.

**GFSA770I OS/390 UNIX REGISTRATION
SUCCESSFUL.**

Explanation: Connection with UNIX established.

System Action: Network File System processing continues.

**GFSA771I(<PROCNAME>) HFS MOUNTS
SUSPENDED.**

Explanation: Hierarchical file system (HFS) mount processing has been suspended by the **freeze=onhfs** operand of the **modify** command. This message is displayed in response to either the **freeze=onhfs** or **status** operands of the **modify** command. Further HFS mount requests from the network are ignored. Existing mounts are unaffected.

System Action: Network File System processing continues.

GFSA772I(<PROCNAME>) HFS MOUNTS RESUMED.

Explanation: Mount requests to hierarchical file system (HFS) file systems have been enabled. Given in response to a console **modify** command.

System Action: Network File System processing continues.

GFSA776I HFS CANNOT RESOLVE PATH NAME
text.

Explanation: There was a failure to resolve a path name with UNIX when initializing from the mount handle data sets. The HFS file system has been removed or renamed. If the user attempts to access a file object under this mount point, the Network File System error response NFSERR_STALE is returned.

System Action: Network File System processing continues.

GFSA777I SERVICE REQUESTER DOES NOT HAVE SECURITY PRIVILEGE.

Explanation: The client user must be defined to Resource Access Control Facility (RACF) as a user of OpenEdition multiple virtual system (MVS) to access hierarchical file system (HFS) file objects.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Check the System Authorization Facility (SAF) security product user profiles.

GFSA782I(<PROCNAME>) NO ACTIVE HFS MOUNT POINTS.

Explanation: This message is in response to the **list=mounts** operand of the **modify** command and shows that there are no OpenEdition clients connected to the Network File System.

System Action: Network File System processing continues.

GFSA783I(<PROCNAME>) NO ACTIVE HFS DATA SETS.

Explanation: This message is in response to the **list=dsnames** operand of the **modify** command and shows that there are no clients actively accessing hierarchical file system (HFS) data sets.

System Action: Network File System processing continues.

GFSA784I *text1* **RPC OS/390 UNIX ERROR**
VNODE_OP *text2* **RC:** *number1*: *text3*
RSN: *number2* *number3*.

Explanation: The NFS has encountered a problem on an interface call with UNIX. The error was encountered during processing of a *text1* remote procedure call (RPC) call. *text2* is the function called when failure occurred. *number1* is the return code and *text3* is the English description of the return code. The combination of *number2* *number3* represents the reason code as returned by UNIX. See UNIX documentation for a full explanation of OpenEdition MVS reason codes (for example, *z/OS UNIX System Services Programming: Assembler Callable Services Reference, SA22-7803*).

System Action: The NFS continues processing.

System Programmer Response: See the UNIX documentation.

GFSA786I MULTI-COMPONENT LOOKUP
REQUEST FOR PATHNAME *text*
CANNOT BE RESOLVED.

Explanation: The multi-component **lookup** request for the path name failed. Possible reasons are:

- Symbolic links cannot be embedded in a multi-component path name as it is not supported at this time
- Pathname specified is either not supported or access is not allowed
- Public path is not set up on this server.

System Action: The request fails. NFS processing continues.

User Response: Construct a different **lookup** request with a valid path name.

GFSA801I MOUNT FAILED: *text*

Explanation: The text can be:

- FILE MAPPING ENABLED BUT NO SIDE FILE SPECIFIED
- STORAGE LIMIT REACHED LOADING MAPPING SIDE FILE
- MAPPING SIDE FILE NOT FOUND
- ERROR OPENING/READING MAPPING SIDE FILE
- MAPPING SIDE FILE HAS INVALID SYNTAX OR FORMAT
- SIDE FILE SPECIFIED BUT MAPPING IS DIS-ALLOWED BY INSTALLATION

User Response:

1. Specify a side file if fileextmap is ON, or
2. Fix problem with mapping side file, or
3. Consult the system administrator to change sfgmax value and re-issue the **mount** command.

GFSA802E REMOUNT FAILED — PHYSICAL FILE SYSTEM CHANGED. PATH: *pathname* PREV: *datasetname* CURRENT: *datasetname*

Explanation: During restart of the NFS server, rebuild of the mount point recorded in the Mount Handle Database failed. The failure occurred because the physical file system for the mount point was changed by a **tso unmount** since the mount point was originally mounted.

System Action: NFS server processing continues.

GFSA811I CANNOT FLUSH PARTIAL RECORDS FOR DATA SET *text1(text2)*: FB *h_digits*.

Explanation: There was not enough memory to allocate storage required to flush partial records to the data set at data set close time. The partial record was discarded.

System Action: The data set is closed. Network File System processing continues.

System Programmer Response: Increase the size of the step region. The data set might be incomplete.

GFSA812I FLUSH FAILED: RC *d_digits1* OFFSET *d_digits2* WAS DROPPED FOR DATA SET *text1(text2)*.

Explanation: When the Network File System was trying to flush cached data at data set close time, the error *d_digits1* was detected. The data at offset *d_digits2* was discarded. This error message follows more specific error messages.

System Action: The data set is closed. Network File System processing continues.

System Programmer Response: See the message preceding this message to determine the correct action. The data set might be incomplete. See Table 49 on page 149 for a description of the return code *d_digits1*.

GFSA813I REMOVE FAILED: RC *h_digits* DSN *text1(text2)*.

Explanation: The error *h_digits* was detected when trying to remove member *text2* from PDS *text1*.

System Action: Network File System processing continues.

System Programmer Response: *h_digits* is the return code from the MVS/DFP STOW macro. See *z/OS DFSMS Macro Instructions for Data Sets* for the explanation of STOW return codes.

GFSA814I RENAME FAILED: RC *h_digits* DSN *text1* OLDMEM *text2* NEWMEM *text3*.

Explanation: The error *h_digits* was detected when trying to rename member *text2* to *text3* in partitioned data set *text1*.

System Action: Network File System processing continues.

System Programmer Response: *h_digits* is the return code from the MVS/DFP STOW macro. See *z/OS DFSMS Macro Instructions for Data Sets* for the explanation of STOW return codes.

GFSA815I RENAME FAILED: RC *h_digits1* OLDDSN *text1* VOL *text2* UNIT *h_digits2* NEWDSN *text3*.

Explanation: The error *h_digits1* was detected when trying to rename the non-VSAM data set *text1* to *text3*. *text2* is disk volume serial number. *h_digits2* is the disk device type code.

System Action: Network File System processing continues.

System Programmer Response: *h_digits1* is a composite of one or more error return codes encountered when trying to rename the data set. The hexadecimal digits are decoded using the list below.

Renaming a data set requires three steps: Uncatalog the old data set name, rename the data set in the disk VTOC, and catalog the new data set name. Should an error occur in either of the last two steps, the prior steps are undone to preserve the old data set name.

Find the step that failed by matching the value in byte zero of the return code with the values under the heading **byte 0** below. Byte three contains the return code from the first failing MVS/DFP service (Uncatalog/catalog or DADSM rename). If further errors occur when trying to recatalog or rename the data set back to the old name, the return codes are placed in bytes one and two respectively.

Byte 0 Meaning/Other Bytes

- | | |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 00 | Error uncataloging old data set name. Byte 3: Uncatalog return code. |
| 01 | Error renaming the data set. Byte 1: Recatalog return code for old data set name. Byte 3: DADSM rename return code. |
| 02 | Error cataloging new data set name. Byte 1: Recatalog return code for old data set name. Byte 2: DADSM rename return code for old data set name. Byte 3: Catalog return code for new data set name. |

Catalog and DADSM rename return codes are documented in *z/OS DFSMSdfp Advanced Services*. Message IEC614I is written for DADSM rename errors

and contains further diagnostic codes. These codes are documented in *z/OS DFSMSdfp Diagnosis Reference*.

GFSA816I HOST NAME OF IP ADDRESS (*d_digits*) WAS NOT FOUND BY TCP/IP.

Explanation: The client host name of IP address *d_digits* is not defined in either the TCP/IP Domain Name Server or the TCP/IP Site Table. See *TCP/IP for MVS: Customization and Administration Guide*.

System Action: Network File System processing continues. The dotted IP address is used as the host name.

System Programmer Response: Insert this client host's entry into either the TCP/IP Domain Name Server or the TCP/IP Site Table.

GFSA817I *text1* REQUEST NOT VALID ON ALIAS NAME *text2*.

Explanation: *text1* can be remove or rename. *text2* is an MVS access method services alias name of a file which also has a true name. Remove (**rm** or **rmdir**) and **rename** (**mv**) requests cannot be run using an alias name. The true file name is required.

System Action: The request is stopped. An I/O error indication is returned to the client. Network File System processing continues.

System Programmer Response: Inform the client user of this error.

User Response: Provide the true name of the file on the request.

GFSA818I(<PROCNAME>) EXPORTS: NO VALID HOST NAMES IN *text* LIST.

Explanation: None of the client host names in the read/write or access list (see *text*) are defined to the network.

System Action: The Network File System does not export the associated directory if the result is a null access list. If the result is a null read/write list, the directory is exported read-only.

System Programmer Response: Correct the host names in the exports data set or have the host names defined to the network.

GFSA819I DATA SET *text1* CREATION USING DATA CLASS = *text2*.

Explanation: Data set *text1* is being allocated using the attributes in data class *text2*.

System Action: Network File System processing continues.

GFSA820I CATALOG ERROR OCCURRED WHILE [RETRIEVING[UPDATING] CATALOG INFORMATION FOR *text*. RETURN CODE IS *d_digits1*, REASON CODE IS *cc-d_digits2*.

Explanation: Catalog Management module IGG0CLcc returned this return code, *d_digits1*, and reason code, *d_digits2*, as the result of a catalog error or an exception condition. *text* was the name of the data set that the [retrieve|update] operation was performed against.

System Action: Network File System processing continues.

System Programmer Response: See message IDC3009I in the *z/OS MVS System Messages, Vol 5*, *z/OS MVS System Messages, Vol 6*, and *z/OS MVS System Messages, Vol 7* manuals for specific return code and reason code information.

GFSA821I ERROR OCCURRED WHILE UPDATING THE FORMAT 1 DSCB FOR *text1* ON *text2*. FUNCTION CODE IS *d_digits1*, RETURN CODE IS *d_digits2*, REASON CODE IS *d_digits3*.

Explanation: *text1* is the name of the data set and *text2* is the volume serial number of the volume on which the data set resides. The function code can be one of the following:

- 2 De-serializing the UCB
- 4 De-serializing the DASD volume
- 12 Searching for the UCB
- 16 Serializing the DASD volume
- 20 Reading the DSCB
- 24 Writing the DSCB

System Action: Network File System processing continues.

System Programmer Response: See the following manuals for specific return code and reason code information:

- 2 *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO*, macro UCBPIN.
- 4 *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN*, macro DEQ.
- 12 *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO*, macro UCBLook.
- 16 *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*, macro RESERVE.

- 20 z/OS DFSMSdfp Advanced Services, Return Codes from CVAFDIR.
- 24 z/OS DFSMSdfp Advanced Services, Return Codes from CVAFDIR.

GFSA822I UNABLE TO PERFORM FILE MAPPING BECAUSE NO SIDE FILE SPECIFIED OR LOADED

Explanation: File mapping cannot be performed because a side file was not specified either as a default or at the mount point.

System Action: Operation fails

System Programmer Response: Specify a side file in the attributes data set.

User Response: Specify a side file in the mount command.

GFSA823I(<PROCNAME>) PUBLIC PATH CANNOT BE ESTABLISHED.

Explanation: The public pathname(s) specified in the *public* keyword cannot be established during server startup. This could be because the path is not exported or it does not exist.

System Action: The NFS server startup ends.

Operator Response: Make sure the public pathname exist and is exported if export list checking is enabled. Correct the problem and restart the server.

GFSA827I REQUEST *h_digits* CREDENTIALS ALLOCATION FAILED.

Explanation: An operation to allocate virtual memory for a credentials block was tried, but was unsuccessful. *h_digits* is the request block address.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Increase the size of the step region.

GFSA829I REQUEST *h_digits* INVALID CREDENTIALS FLAVOR *d_digits*.

Explanation: Incorrect credentials type received from client. Probable client software error. The Network File System supports UNIX and non-authentication styles. *h_digits* is the request block address.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Inform the client user that the credentials used are not valid.

GFSA832I REQUEST *h_digits* INVALID MEMBERNAME FOR *text*.

Explanation: *text* is the member name of a partitioned data set (PDS) that was specified as a file name by the Network File System client user. The file name specified was incorrect or not found in the PDS. *h_digits* is the request block address.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Inform the client user of this error.

User Response: Correct the error and resubmit the request.

GFSA833I REQUEST *h_digits* PARSE FAILED FOR *text*.

Explanation: *text* is the member name of a partitioned data set (PDS) or a data set name that was specified as a file name by the Network File System client user. The file name specified was incorrect, not found in the PDS, or was an incorrect or non-existent data set. *h_digits* is the request block address.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Inform the client user of this error.

User Response: Correct the error and resubmit the request.

GFSA840I DYNAMIC ALLOCATION FAILED WITH RETURN CODE *h_digits* FOR DATA SET *text1(text2)*.

Explanation: Dynamic file allocation error. *h_digits* is the dynamic allocation return code. *text1* is the data set name. *text2* is the member name if any.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: This message is preceded by either message GFSA853I or GFSA854I. See the programmer response for the message that precedes this message to determine the appropriate action.

GFSA842I *text1* UNSUPPORTED DSORG *text2*.

Explanation: The data set organization of data set *text1* is not supported by the Network File System. *text2* can be the characters "ISAM" or "UNKNOWN"

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: Inform the client user of this error.

GFSA843I CREATE FAILED FOR *text*.

Explanation: An error occurred while the Network File System was trying to create the data set *text*. This message follows other messages that describe the error in greater detail.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: See the message(s) preceding this message to determine the appropriate response.

GFSA847I IDCAMS ERROR: *text*.

Explanation: *text* is an access method services error message.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: See *MVS System Messages* for more information on the access method services error message, or use **LookAt**. For a description of LookAt, see "Using LookAt to look up message explanations" on page xv.

GFSA848I PDS *text* IS NOT EMPTY.

Explanation: The Network File System client user issued a **rmdir** (remove directory) AIX or UNIX command to remove a partitioned data set that was not empty. The Network File System version 2 protocol specification requires the directory (PDS) to be empty before it is removed. This is a Network File System client user error. *text* is the name of the PDS.

System Action: The request is stopped. An error is returned to the client. Network File System processing continues.

System Programmer Response: Inform the Network File System client user of this error.

User Response: Remove all files in the directory then resubmit the **rmdir** request.

GFSA849I NEW AND OLD FILES ARE NOT MEMBERS OF THE SAME PDS.

Explanation: Rename is not allowed for a member of a PDS when the target name is not in the same PDS. This is a Network File System client user error.

System Action: The request is stopped. An error is returned to the client. Network File System processing continues.

System Programmer Response: Inform the Network File System client user of this error.

User Response: Check the filename used. Correct it and try again.

GFSA853I DYNAMIC ALLOCATION: INPUT VALIDATION ROUTINE REJECTED ALLOCATION.

Explanation: Dynamic allocation failed when issued by the installation input validation routine. Probable installation configuration errors.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: See the *z/OS MVS Installation Exits* for more information on the Input Validation routine.

GFSA854I DYNAMIC ALLOCATION: RC=*d_digits* ERROR=*h_digits1* INFO=*h_digits2* : *text*.

Explanation: Dynamic allocation failed with return code *d_digits*, error reason code *h_digits1* and information reason code *h_digits2*. *text* is a description of the interrupted dynamic allocation request.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: See the *z/OS MVS Programming: Authorized Assembler Services Guide* for more information about these codes.

GFSA858I OPEN FAILED RC *d_digits* FOR DATA SET *text1(text2)*.

Explanation: An error occurred while trying to open the data set. *text1* is the data set name. *text2* is the member name (if any). *d_digits* is the return code.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: If this error was not caused by an out-of-memory condition, contact your programming support personnel. See Table 49 on page 149 for a description of the return code *d_digits*.

GFSA859I READ FAILED RC *d_digits* FOR DATA SET *text1(text2)*.

Explanation: An error occurred while trying to read the data set. *text1* is the data set name. *text2* is the member name (if any). *d_digits* is the return code.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: If this error was not caused by an out-of-memory condition, contact your programming support personnel. See Table 49 on page 149 for a description of the return code *d_digits*.

GFS A860I WRITE FAILED RC *d_digits* FOR DATA SET *text1(text2)*.

Explanation: An error occurred while trying to write the data set. *text1* is the data set name. *text2* is the member name (if any). *d_digits* is the return code.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: If this error was not caused by an out-of-memory condition, contact your programming support personnel. See Table 49 on page 149 for a description of the return code *d_digits*.

GFS A862I CATALOG (*text*) COULD NOT BE LOCATED.

Explanation: The user catalog named *text* that contains the entry for an index could not be located. The catalog does not exist or is not mounted. If it does not exist, the entry in the master catalog might be incorrect.

System Action: Network File System processing continues.

System Programmer Response: Investigate why the catalog could not be found and take corrective action.

GFS A863I READDIR ON ROOT IS NOT ALLOWED.

Explanation: The user tried to list the contents of the master catalog.

System Action: The request is stopped. Network File System processing continues.

GFS A864I(<PROCNAME>) CANNOT OPEN THE EXPORTS DATA SET.

Explanation: The server was unable to open the exports data set defined in the job control language (JCL) for DDNAME EXPORTS. The DD statement might be missing or the data set name might be incorrect.

System Action: The Network File System stops.

Operator Response: Notify your system programmer.

System Programmer Response: Correct the JCL for DDNAME EXPORTS.

GFS A865I(<PROCNAME>) EXPORTS: UNEXPECTED OPTION (*text*)-- SHUTDOWN SCHEDULED.

Explanation: The option information provided in *text* is incorrect. This error could occur as a result of unexpected blanks, incorrect syntax, or mutually exclusive options (for example, both *ro* and *rw*).

System Action: Checking of the exports data set continues but shutdown of the Network File System occurs at its completion.

System Programmer Response: Correct the exports data set and restart the Network File System.

GFS A866I(<PROCNAME>) EXPORTS: DIRECTORY *text* WAS NOT EXPORTED.

Explanation: An error was encountered severe enough to prevent the data set or index named *text* from being exported. This message follows a more specific error message.

System Action: Network File System processing continues.

System Programmer Response: Correct the exports data set.

GFS A867I(<PROCNAME>) EXPORTS: *text1* CANNOT BE EXPORTED BECAUSE *text2* ALREADY IS.

Explanation: The data set or index named in *text1* is a parent directory or a subdirectory of the data set or index named in *text2* which is already exported.

System Action: Network File System processing continues.

System Programmer Response: Correct the exports data set.

GFS A868I(<PROCNAME>) EXPORTS: HOST (*text*) WITH UNKNOWN ADDRESS TYPE (*d_digits*) HAS BEEN IGNORED.

Explanation: The host named *text* has an IP address (*d_digits*) that is not valid.

System Action: Network File System processing continues.

System Programmer Response: Correct the exports data set.

GFS A869I(<PROCNAME>) EXPORTS: UNKNOWN HOST (*text*) HAS BEEN IGNORED.

Explanation: The client host named *text* is not defined to the network. If there are other defined client hosts specified for the option, the Network File System ignores this undefined client host as if had not been specified. (However, see message GFS A818I for special handling of null lists.)

System Action: Network File System processing continues.

System Programmer Response: Correct the host name in the exports data set or have this host name defined to the network.

GFSA8711 REQUEST *h_digits* HAS MISMATCHED
UID: CRED = *d_digits1* ARGS =
d_digits2.

Explanation: *h_digits* is a block created for the logon or logout request. *d_digits1* represents the credential user ID number. *d_digits2* represents the client user ID number. The user ID numbers do not match and this is considered a security failure.

System Action: The client logon or logout request is stopped. Network File System processing continues.

System Programmer Response: This is a Network File System client application problem. If the Network File System client application is offered by IBM, inform the IBM programming support center. If the Network File System client application is offered by your programming support personnel, contact your programming support personnel.

GFSA8761 I/O ERROR ON DSN = *text1(text2)*
SENSE *h_digits1* IOBCSW *h_digits2*
h_digits3. ACCESS METHOD RC =
h_digits4 ACCESS METHOD RSN =
h_digits5

Explanation: The physical I/O layer tried to check some previous operation in the data set and the check failed. *text1* is the data set name. *text2* is the member name (if any). *h_digits1* is the sense bytes 0 and 1 from the device. *h_digits2* is the first 3 bytes of the channel status word from the device. *h_digits3* is the last 4 bytes of the channel status word from the device. *h_digits4* is the access method return code. *h_digits5* is the access method reason code.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: See the appropriate device documentation for more information on the sense bytes and channel status word.

GFSA8771 R0=*h_digits1* R1=*h_digits2: text* ACCESS
METHOD RC = *h_digits3* ACCESS
METHOD RSN = *h_digits4*

Explanation: A SYNAD error was detected during physical I/O operation. *h_digits1* is the contents of register 0; *h_digits2* is the contents of register 1. The *text* is the message returned from the SYNAD analysis function macro. *h_digits3* is the access method return code. *h_digits4* is the access method reason code.

System Action: The request is stopped. If the error detected is a B37, D37 or E37 abend, the Network File System restores the file size to the last known file size before the SYNAD error. Network File System processing continues.

System Programmer Response: A Data Management message should have been displayed on the console. See *MVS System Messages* for a

description of the return code to determine the corrective action or use LookAt. For a description of LookAt, see "Using LookAt to look up message explanations" on page xv.

**GFSA8781(<PROCNAME>) CANNOT OPEN THE
CHKLIST DATA SET.**

Explanation: The Network File System cannot open the checklist data set, specified by the CHKLIST DD statement in the MVS NFS startup procedure.

System Action: The Network File System ends.

Operator Response: Contact the system programmer.

System Programmer Response: Check for validity of the specified data set. Correct the JCL for CHKLIST DD statement in the MVS NFS startup procedure.

**GFSA8791(<PROCNAME>) PROBLEMS
ENCOUNTERED PARSING THE
CHKLIST DATA SET.**

Explanation: The Network File System was not able to successfully parse the checklist data set specified by the CHKLIST DD statement.

System Action: The Network File System ends.

Operator Response: Contact the system programmer.

System Programmer Response: Correct the content of the checklist data set.

GFSA8811 FUB: *h_digits1* UNABLE TO ACCESS
FILE *text1(text2)* OWNED BY FUB
h_digits2.

Explanation: A user has tried to access a data set already locked internally by the Network File System for writing by another user. The data set has not been released yet. *h_digits1* is the address of the File Usage Block. *text1* is the data set name. *text2* is the member name (if any). *h_digits2* is the address of the File Usage Block which currently has the file allocated.

System Action: The request is stopped. A "Not Owner" error message is returned to the user. Network File System processing continues.

System Programmer Response: If queried by the user, the "Not Owner" error message, as it relates to the Network File System, is described in *z/OS Network File System User's Guide*.

GFSA8861 UNABLE TO WRITE RECNO *d_digits* TO
text1(text2) DUE TO PREVIOUS ERROR.

Explanation: An error was previously detected which prevents writing to a data set.

System Action: The request is stopped. Network File System processing continues.

System Programmer Response: If the previous error cannot be determined, contact your programming support personnel.

GFSA895I REQUEST *h_digits* - FILE *text* NOT ALLOCATED.

Explanation: The Network File System did not have the data set *text* open on a request to close the file. The file name might have been specified incorrectly, or the time out might have already occurred for this data set, causing the server to close the data set.

In the message text:

h_digits The request block address.

text The data set name.

System Action: The system ends the request. Return code X'131' is passed back to the user. The system continues processing for the Network File System.

GFSA896I REQUEST *h_digits1* - FILE BLOCK *h_digits2* ASSOCIATED WITH FILE *text* NOT IN USE BY CREDENTIALS *h_digits3*.

Explanation: The request to close file *text* was received, but the file was not opened by the client. A file can only be closed by the same client that opened the file. *h_digits1* is the request block address. *h_digits2* is the file block address. *h_digits3* is the credentials block address.

System Action: The request is stopped. Return code 132 is passed back to the client. Network File System processing continues.

GFSA897I RECORD *d_digits1* SIZE *d_digits2* IS *text1*: *text2* = *d_digits3* DSN = *text3*(*text4*).

Explanation: The record received from the client can be too long or too short. *d_digits1* is the record number. *d_digits2* is the record size. *text1* can be the characters "TOO LONG" or "TOO SHORT". *text2* can be the characters "MAXIMUM" or "MINIMUM". *d_digits3* shows the maximum or minimum length allowed. *text3* is the data set name. *text4* is the member name (if any).

System Action: The request is stopped. Network File System processing continues.

GFSA898I EOL SEQUENCE MISMATCH FOR DATA SET *text1*(*text2*).

Explanation: The end-of-line terminator was not found in the same place as the previous end-of-line terminator for an offset that is being re-written by the client. This is a Network File System client error.

System Action: The request is stopped. Network File System processing continues.

User Response: Make sure that the record you are writing is the same size as the record you are replacing.

GFSA899I BLANKSTRIP MODE: TRAILING BLANK(S) IN RECORD *d_digits* IS NOT ALLOWED DSN = *text1*(*text2*).

Explanation: Writing data in text mode with blank stripping enabled and blanks at the end of the line to a data set with fixed-length records is not allowed.

System Action: The request is stopped. Network File System processing continues.

GFSA900I(<PROCNAME>) MOUNT PROCESSING ACTIVE.

Explanation: Mounts can be issued from the network. This message is displayed in response to the **status** operand of the **modify** command.

System Action: Network File System processing continues.

GFSA901I(<PROCNAME>) MOUNT PROCESSING SUSPENDED.

Explanation: Mount processing has been suspended by the **freeze=on** operand of the **modify** command. This message is displayed in response to either the **freeze=on** or **status** operands of the **modify** command. Further mount requests from the network are ignored. Existing mounts are unaffected.

System Action: Network File System processing continues.

GFSA902I(<PROCNAME>) MOUNT PROCESSING RESUMED.

Explanation: Mount processing has been resumed by the **freeze=off** operand of the **modify** command. Further mount requests from the network are be honored. Existing mounts are unaffected.

System Action: Network File System processing continues.

GFSA903I(<PROCNAME>) TASK *h_digits1* TCB *h_digits2* PROGRAM = *text1* = *text2*.

Explanation: *h_digits1* is the task queue address. *h_digits2* is the task control block address. *text1* is the program name. *text2* is the specific task name. This is in response to the **status** operand of the **modify** command.

System Action: Network File System processing continues.

GFSA904I(<PROCNAME>) HFS MOUNT PROCESSING ACTIVE.

Explanation: Mounts can be issued from the network. This message is displayed in response to the **status** operand of the **modify** command.

System Action: Network File System processing continues.

GFSA908I(<PROCNAME>) HFS PROCESSING DISABLED.

Explanation: HFS processing is suspended. This message is displayed in response to the status operand of the **modify** command.

System Action: The NFS processing continues.

GFSA909E(<PROCNAME>) UNMOUNT COMMAND FAILED: MOUNT POINT STILL IN USE

Explanation: This message is in response to the **unmount** operand of the **modify** command. The unmount processing fails because the file system is still in use. The user can retry the **unmount** command later after there is no reference to file system.

System Action: The system ends the request. The system continues processing for the NFS.

GFSA910I(<PROCNAME>) text ACTIVE = d_digits.

Explanation: This message is in response to the **list=mounts** operand of the **modify** command and shows the mounted file space (MVS path) and the number of active clients (*d_digits*) mounting those files by that path. The number might not be accurate if a client has crashed without unmounting the files.

System Action: Network File System processing continues.

GFSA911I(<PROCNAME>) text1(text2) ACTIVE = d_digits.

Explanation: This status message is in response to the **list=mounts** operand of the **modify** command and shows the mounted file space (MVS path) and the number of active clients (*d_digits*) mounting those files by that path. The number might not be accurate if a client has crashed without unmounting the files.

System Action: Network File System processing continues.

GFSA912I(<PROCNAME>) text.

Explanation: The data set name appears in response to a **list=dsnames** operand of the **modify** command and shows a currently active data set.

System Action: Network File System processing continues.

GFSA913I(<PROCNAME>) text1(text2).

Explanation: The PDS member appears in response to the **list=dsnames** operand of the **modify** command and shows a currently active data set.

System Action: Network File System processing continues.

GFSA914I(<PROCNAME>) text DEALLOCATED.

Explanation: The data set name *text* appears in response to a **release=dsname(member)** operand of the **modify** command after successful deallocation.

System Action: Network File System processing continues.

GFSA915I(<PROCNAME>) text NOT ALLOCATED.

Explanation: The data set name *text* appears in response to the **release=dsname(member)** operand of the **modify** command if the data set or member specified to be released was not found.

System Action: Network File System processing continues.

GFSA916I(<PROCNAME>) text UNMOUNTED.

Explanation: The data set name *text* appears in response to an **unmount** (data set or member) operand of the **modify** command after a successful unmount.

System Action: Network File System processing continues.

GFSA917I(<PROCNAME>) text NOT MOUNTED.

Explanation: The data set name *text* appears in response to the **unmount** (data set or member) operand of the **modify** command if the data set or member specified to be unmounted was not found in the current mount list.

System Action: Network File System processing continues.

GFSA918I(<PROCNAME>) text IS NOT A VALID DATA SET NAME.

Explanation: The data set name specified on either the **release** or **unmount** operand of the **modify** command is not a valid MVS data set name.

System Action: Network File System processing continues.

Operator Response: Specify the command again with a valid data set name.

GFSA919I(<PROCNAME>) *text* IS NOT A VALID MEMBER NAME.

Explanation: The member name specified on either the **release** or **unmount** operand of the **modify** command is not a valid MVS member name.

System Action: Network File System processing continues.

Operator Response: Specify the command again with a valid member name.

GFSA920I(<PROCNAME>) NO ACTIVE MOUNT POINTS.

Explanation: This message is in response to the **list=mounts** operand of the **modify** command and shows that there are no clients connected to the Network File System.

System Action: Network File System processing continues.

GFSA921I(<PROCNAME>) NO ACTIVE DATA SETS.

Explanation: This message is in response to the **list=dsn** operand of the **modify** command and shows that there are no clients actively accessing data sets.

System Action: Network File System processing continues.

GFSA922I(<PROCNAME>) VERIFY: (*text*) IS NOT A VSAM DATA SET.

Explanation: This message is in response to the **verify** operand of the **modify** command and shows that the data set name specified is not a virtual storage access method (VSAM) data set.

In the message text:

text The data set name.

System Action: The system ends the request. The system continues processing for the Network File System.

GFSA923I(<PROCNAME>) VERIFY SUCCESSFUL FOR (*text*).

Explanation: This message is in response to the **verify** operand of the **modify** command. It shows that the verify was successful for the VSAM data set.

In the message text:

text The data set name.

System Action: The system continues processing for the Network File System.

GFSA924I(<PROCNAME>) VERIFY FAILED WITH RC = *d_digits* FOR (*text*).

Explanation: This message is in response to the **verify** operand of the **modify** command. It shows that the verify failed with a return code for the data set. Message GFSA847I follows this message in the log data set.

In the message text:

d_digits The return code.

text The data set name.

System Action: The system ends the request. The system continues processing for the Network File System.

GFSA925I(<PROCNAME>) ERROR WAS DETECTED IN THE EXPORTS FILE. EXPORT LIST NOT REBUILT.

Explanation: This is the reply from the **exportfs** operand of the **modify** command, indicating that one or more errors were detected in the exports data set; for example, the exports data set cannot be opened.

System Action: Network File System processing continues. The existing exports list is not changed.

Operator Response: Notify the system programmer.

System Programmer Response: Review previous console error messages for detailed information as to the specific error in the exports data set.

GFSA926I(<PROCNAME>) EXPORT LIST HAS BEEN REBUILT SUCCESSFULLY.

Explanation: This is the reply from the **exportfs** operand of the **modify** command, indicating that the command has completed normally.

System Action: Network File System processing continues with the new exports data set in effect.

System Programmer Response: Network File System processing continues with the new exports data set in effect.

GFSA927I(<PROCNAME>) MODIFY EXPORTS COMMAND IGNORED - THE EXPORTS FILE IS NOT BEING USED FOR SECURITY CHECKING.

Explanation: This is the reply from the **exportfs** operand of the **modify** command, indicating that the command was ignored because the site attribute for security requested that no exports file checking be done.

System Action: Network File System processing

continues with the security options unchanged.

GFSA930I(<PROCNAME>) LOG DATA SET IS SWITCHED FROM *text1* TO *text2*.

Explanation: A “no space” or an I/O error condition is detected when writing to the log data set. Network File System logging is now switched to the other log data set. *text1* is the DD associated with the switched-from log data set. *text2* is the DD associated with the switched-to log data set.

System Action: Network File System processing continues.

Operator Response: If requested by the installation, back up the switched-from log data set at this point. The switched-from data set is reused when the switched-to log data set is also filled.

System Programmer Response: Consider allocating larger NFS server log data sets for future Network File System usage. Note that the last data buffer is lost when the log is switched.

GFSA931I(<PROCNAME>) NETWORK FILE SYSTEM SERVER LOGGING IS TERMINATED.

Explanation: The Network File System logging is ended. This can be caused by a “no space” condition of the log data set.

System Action: Network File System processing continues.

System Programmer Response: Allocate a larger log data set for future Network File System usage.

GFSA932I(<PROCNAME>) LOG DATA SET *text* IS FLUSHED.

Explanation: The data buffer of the active log data set is flushed to disk. If the log is written to standard error, *text* is “STDERR”. Otherwise *text* is the associated DD of the active log data set.

System Action: Network File System processing continues.

GFSA933I(<PROCNAME>) LOG DATA SET IS RE-INITIALIZED.

Explanation: The log data set is re-initialized.

System Action: Network File System processing continues.

GFSA934I(<PROCNAME>) NFSLOG1 OR NFSLOG2 DD STATEMENTS NOT DEFINED.

Explanation: The “NFSLOG1” or the “NFSLOG2” DD statements are not coded in the Network File System start-up cataloged procedure.

System Action: The server stops.

System Programmer Response: Code the NFSLOG1 and NFSLOG2 DD statements and allocate the associated log data sets.

GFSA935I(<PROCNAME>) SWITCHED-TO LOG IN ERROR. NETWORK FILE SYSTEM SERVER LOGGING CONTINUED ON DD:*text*.

Explanation: An operator issued a **switchlog** command but an open error is detected in new log data set. Network File System logging is continued in the original log data set. *text* is the DD associated with the original log data set.

System Action: NFS processing continues.

Operator Response: Fix the non-active log data set.

GFSA936I(<PROCNAME>) NETWORK FILE SYSTEM SERVER LOG *text* SET TO FORCELOG.

Explanation: The Network File System server log was closed to force all log data to disk immediately.

System Action: The Network File System continues.

GFSA949I COMMAND *text* NOT VALID.

Explanation: The operator has entered the **operator** command or operand *text* which is not valid.

System Action: Network File System processing continues.

Operator Response: Enter the operand of the **modify** command again with the correct syntax.

GFSA950I Unknown flag '*character*'

Explanation: The character *character* specified on the **mvslogin**, **mvslogout**, or **showattr** command is not a valid option. A usage message might follow this message.

User Response: See *z/OS Network File System User's Guide* for a description of the valid options used with the command.

GFSA951I *text* : can't find name for uid *d_digits*.

Explanation: There was an error reading information for UID *d_digits* from the *etc/passwd* file.

User Response: Correct the *etc/passwd* file and try the command again.

GFSA952I Retyped password does not match

Explanation: The password entered when message GFSA975I was displayed does not match the password entered when message GFSA974A was displayed.

User Response: Start the **mvlogin** command sequence again.

GFSA953I Password change required by host.

Explanation: The multiple virtual system (MVS) password for the user ID passed to the host has expired. A new password is required. Message GFSA974I follows this message.

GFSA954I Host *text1* returned error *d_digits: text2*

Explanation: An error was detected during **mvlogin** processing. Host *text1* returned error code *d_digits* and message *text2* to the client.

User Response: The password or user ID might be incorrect. Start the **mvlogin** command sequence again and use the correct password or user ID.

GFSA955I *text* logged in ok.

Explanation: The MVS user ID *text* was logged in without any errors.

GFSA956I usage: *text* [-pn][-g group][-a account] hostname [mvs_username]

Explanation: Usage information for the **text** command.

User Response: Enter the command using the correct syntax.

GFSA957I Host *text1* returned error *d_digits: text2*

Explanation: An error was detected during **mvlogout** processing. Host *text1* returned error code *d_digits* and message *text2* to the client.

User Response: Notify your MVS system programmer.

GFSA958I uid *text* logged out ok.

Explanation: The MVS user ID *text* was logged out successfully.

GFSA959I usage: *text* hostname

Explanation: Usage information for the *text* command.

User Response: Enter the command using the correct syntax.

GFSA960I *text1*: host "*text2*" unknown.

Explanation: The host *text2*, specified on command *text1*, is not known to the network.

User Response: Correct the host name specified and try the command again.

GFSA961I *text1*: *text2*

Explanation: Command *text1* received an error when trying to create a client transport handle using the **clntudp_create** TCP/IP remote procedure call. *text2* is the message produced by the **clnt_pcreateerror** TCP/IP procedure. This message is issued if:

- The host name is unknown
- The host is not operational
- The Network File System on the named host is not operational

User Response: The user response is:

- Correct the host name specified and try the command again,
 - Make sure the specified host is operational and try the command again, or
 - Make sure the NFS server on the named host is operational and try the command again.
-

GFSA964I *text*: Error: cannot determine server.

Explanation: The *text* command found the mount path, but the server name was not returned by the local operating system service that keeps mount point information.

User Response: Correct the mount point table and try the command again.

GFSA965I *text1*: Error: *text2* mounted from server *text3*, not *text4*.

Explanation: The wrong host name was specified for the *text1* command. *text2* is mounted from server *text3* instead of server *text4*.

User Response: Specify the command again with the correct host name.

GFSA966I *text*: Error: unknown return from usage routine.

Explanation: The usage routine used for the **text** command returned an unknown error code.

User Response: Contact your programming support personnel.

GFSA967I Host Error: *text*.

Explanation: The host returned an error and message *text*. This might be due to:

- A porting failure occurred for the **showattr** command.
- The Network File System is not compatible with the **showattr** command on the client.

User Response: Contact your programming support personnel.

GFSA968I Error: Drive *text* not mounted.

Explanation: The drive *text* was not mounted. The **showattr** command cannot show the attributes for this drive.

User Response: Mount the drive and reissue the command.

GFSA969I Error: Can't open *text* for read.

Explanation: The file *text* could not be opened to read the mount path.

User Response: Correct the file and reissue the command.

GFSA970I Error: Directory *text* not mounted.

Explanation: The directory *text* was not mounted. The **showattr** command cannot show the attributes for this directory.

User Response: Mount the directory and reissue the command.

GFSA971I Error: filesystem *text* is local.

Explanation: The file system *text* is not a Network File System file system. The **showattr** command is for Network File System file systems only.

User Response: Reissue the command for a Network File System file system.

GFSA972I usage: *text1* [-t] hostname [*text2*]

Explanation: Usage information for the *text1* command. *text2* is the operating system dependent mount point format.

GFSA973A Enter MVS password:

Explanation: The Network File System requires a password for the user.

User Response: If an MVS user ID was specified on the **mvslogin** command, enter the password for that MVS user ID. If no MVS_user ID was specified, the name from *etc/passwd* for the UID that issued the **mvslogin** command was passed to the Network File System. Enter the MVS password for this user.

GFSA974A Enter new MVS password:

Explanation: The MVS password for the user ID passed to Network File System has expired.

User Response: Enter a new MVS password.

GFSA975A Retype new MVS password:

Explanation: MVS requires the new password to be entered twice for verification.

User Response: Enter the new MVS password again.

GFSA976I *text1: text2*

Explanation: Command *text1* received an error when trying to create a client transport handle using the *clnt_call* TCP/IP remote procedure call. *text2* is the message produced by the *clnt_perror* TCP/IP procedure.

User Response: Contact your system administrator.

GFSA977I *text*:

Explanation: Command *text* received an error when trying to create a client transport handle using the *clnt_call* TCP/IP remote procedure call (RPC). This message is followed by the message produced by the *clnt_perror* TCP/IP procedure.

User Response: Contact your system administrator.

GFSA978I *text* logged in ok. Mismatch in uid/gid:
OpenEdition uid is *digit_1*, gid is
digit_2, client uid is *digit_3*, gid is *digit_4*

Explanation: The OpenEdition UID/GID does not match the client machine UID/GID. The authentication is successful and the message is for informational use only.

**GFSA991E(<PROCNAME>) MESSAGE FORMAT
FROM USER EXIT ROUTINE(S) IS
INCORRECT. USER *text* EXIT
ROUTINE(S) HAS(HAVE) ENDED.**

Explanation: The *test* exit routine returned an incorrect length message. *text* can be either LOGON AND FILE SECURITY or FILE SECURITY.

System Action: The NFS server processing continues without the user exit routine(s).

Operator Response: Record MVS operator console message and notify the NFS system programmer.

System Programmer Response: Correct user exit routine(s), relink the user exit routine(s), and restart the NFS server.

**GFSA996E(<PROCNAME>) INSTALLATION
DEFAULT TRANSLATION TABLE
CANNOT BE INITIALIZED.**

Explanation: The Network File System ends because one of these conditions happens during Network File System startup:

- The NFSXLAT DD statement is not coded in the Network File System start-up catalog procedure, and the **xlat** processing attribute is specified.
- Translation table data set defined in the NFSXLAT DD statement is not a PDS or PDSE.
- translation table specified in the translation table data set cannot be found.
- Translation table contained in the translation table data set is in an incorrect format.

System Action: Network File System processing stops.

System Programmer Response: Do the following before restarting the Network File System:

- Code the NFSXLAT DD statement in the Network File System start-up catalog procedure correctly.
- Allocate the translation table data set defined in the NFSXLAT DD statement.
- Make sure the translation table in the translation table data set exists.
- Make sure the translation table specified in the translation table data set is in the correct format.

unsuccessful. *text* is the member name that contains the translation table. The cause might be:

- The NFSXLAT DD statement is not coded in the Network File System start-up catalog procedure.
- The NFSXLAT DD statement is not coded correctly in the Network File System start-up catalog procedure.
- Translation table data set defined in the NFSXLAT DD statement is not a PDS or PDSE.
- translation table specified in the translation table data set cannot be found.
- The format of the translation table contained in the translation table data set is not valid.

System Action: Network File System processing stops.

User Response: This error occurs on mount operation. Contact your system programmer for the correct member name for the translation table.

GFSA998I TRANSLATION TABLE *text* IS LOADED.

Explanation: Network File System loaded the translation table, *text*, successfully.

System Action: Network File System processing continues.

**GFSA997I ERROR IN READING TRANSLATION
TABLE, *text*.**

Explanation: The Network File System tried to read the translation table, *text*, during the mount but was

Client messages

This is a listing of the messages generated by the NFS client. A message, explanation, and recommended action is supplied where applicable. Data is substituted for any part of a message shown here in *italics*.

Messages that appear on the MVS operator's console for the NFS client will be in the following example format: **GFSC700I OS/390 VERSION 1 NETWORK FILE SYSTEM CLIENT(HDZ11TC) STARTED**

Table 42 shows the NFS client MVS operators console message format:

Table 42. NFS client MVS operators console message format

GFSC	Component identifier for the NFS client.
700	A unique message number.

Table 42. NFS client MVS operators console message format (continued)

I	Message type:
A	Action; the user must perform a specific action.
E	Eventual action; the user must perform an action when time is available.
I	Informational; no user action is required.
OS/390 VERSION 1 NETWORK FILE SYSTEM CLIENT(HDZ11TC) STARTED	Message text.

Messages appear in the NFS client log data set in the same format as the following example: **12:34:18 GFSC100E (E) CCXDR 11 XDR_DISP: RPC REQUEST (7F638E30) FAILED, RETURN VALUE -1 RETURN CODE 0000467 REASON CODE 5 (TIMED OUT)**

Table 43 shows the NFS client client log data set message format:

Table 43. NFS client log data set message format

12:34:18	The time stamp (hours:minutes:seconds).
GFSC	Component identifier for the NFS client.
100	A unique message number.
E	Message type:
A	Action; the user must perform a specific action.
E	Eventual action; the user must perform an action when time is available.
I	Informational; no user action is required.
(E)	The message level: E (error), W (warning), or I (informational).
CCXDR 11	'CCXDR' is the last 5 characters of CSECT name. '11' is the message sequence within the function.
XDR_DISP	The first 8 characters of the function name.
RPC REQUEST (7F638E30) FAILED, RETURN VALUE -1 RETURN CODE 0000467 REASON CODE 5 (TIMED OUT)	Message text.

The messages are listed in numerical order (the time stamp, message level, and programming support information are not shown).

Notes:

1. *h_digits* denotes hexadecimal digits, and *d_digits* denotes decimal digits. *text* represents variable text (such as a data set name).
2. Messages GFSC098I and GFSC099I are intended for IBM support personnel when they are performing diagnosis.

Table 44 shows the common variables in the message text:

Table 44. Common variables

Variable	Meaning
retv	decimal return value
retc	decimal return code
rsnc	decimal reason code
returncd	8-digit hexadecimal return code
reasoncd	8-digit hexadecimal reason code
h_digit	8-digit hexadecimal address
d_digit	decimal digits
text	Place holder for long text of different lengths

Table 45 can be used for initial translation of the reason code, *reasoncd*, information presented in messages related to parsing errors.

Table 45. Parsing error (when reason code is 6E01xxxx)

Last 4 hex digits of <i>reasoncd</i>	Reason
Fxxx	Unknown keyword
11yy	Host name
12yy	Path name
13yy	Keyword acdirmax
14yy	Keyword acdirmin
15yy	Keyword acregmax
16yy	Keyword acregmin
17yy	Keyword cln_ccsid
18yy	Keyword srv_ccsid
19yy	Keyword hard
1Ayy	Keyword soft
1Byy	Keyword retrans
1Cyy	Keyword timeo
1Dyy	Keyword wsiz
1Eyy	Keyword rsiz
1Fyy	Keyword retry
2Ayy	Keyword vers
2Byy	Keyword proto
21yy	Keyword Biod
22yy	Keyword Bufhigh
23yy	Keyword DelayWrite
24yy	Keyword ReadAhead
25yy	Keyword AttrCaching
26yy	Keyword DataCaching
27yy	Keyword DynamicSizeAdj
28yy	Keyword delim
29yy	Keyword xlat

Table 45. Parsing error (when reason code is 6E01xxxx) (continued)

Last 4 hex digits of <i>reasoncd</i>	Reason
Notes:	
xxx	Offset to the beginning of the mount parameter to the bad keyword
yy	See Table 46 for more details

Table 46 can be used for further translation of the reason code, *reasoncd*, information presented in messages related to parsing errors.

Table 46. Parsing error (when reason code is from 6E0111yy to 6E0129yy)

Last 2 hex digits of <i>reasoncd</i>	Reason
01	Null host name or null path name
02	Blank detected
03	Incorrect member name in the path name
04	Missing double quote
05	No member name found
06	Missing left parenthesis
07	Incorrect number
08	Number is larger than 2G
09	Incorrect multiplier, must be K, M, or G
0A	Missing right parenthesis
0B	The specified number is not within the allowable range
0C	Incorrect keyword parameter value
0D	Mutually exclusive keyword/option
0E	Keyword is not allowed in the mount option
0F	Keyword is not allowed in the installation parameter

Table 47 contains the rest of the reason code, *reasoncd*, information presented in messages.

Table 47. Parsing error (when reason code is 6Exxyyyy')

<i>xx</i> of <i>reasoncd</i>	Module	<i>yyyy</i> of <i>reasoncd</i>	Description
02	TCP/IP common error	0001	<i>clntudp_create()</i> failed
02	TCP/IP common error	0002	Server NFS port not 2049
02	TCP/IP common error	0003	<i>authunix_create()</i> failed
02	TCP/IP common error	0004	<i>clnt_control()</i> timeout failed
02	TCP/IP common error	0005	<i>clnt_control()</i> total timeout failed
02	TCP/IP common error	0006	<i>clnttcp_create()</i> failed
02	TCP/IP common error	0007	<i>clntudp_bufcreate()</i> failed
02	TCP/IP common error	<i>1num</i>	<i>clnt_call()</i> timeout (:retc=0467h, ETIMEDOUT)
02	TCP/IP common error	<i>1num</i>	<i>clnt_call()</i> EINTR (:retc=0078h, EINTR)

Table 47. Parsing error (when reason code is 6Exyyyy') (continued)

<i>xx of reasoncd</i>	Module	<i>yyyy of reasoncd</i>	Description
03	TCP/IP error	<i>yyyy</i>	TCP error - <i>clnt_control()</i> failed
03	TCP/IP error	<i>yyyy</i>	TCP error - <i>authunix_create()</i> failed
03	TCP/IP error	<i>yyyy</i>	TCP error - <i>clnt_call()</i> failed
04	TCP/IP error	<i>yyyy</i>	Authentication error <i>authunix_create()</i> failed
05	NFS protocol error	0001	00001, not owner (:retc=0088h, EPERM)
05	NFS protocol error	0002	00002, no such file/directory (:retc=0081h, ENOENT)
05	NFS protocol error	0005	00005, I/O error (:retc=007Ah, EIO)
05	NFS protocol error	0006	00006, no such device or address (:retc=008Ah, ENXIO)
05	NFS protocol error	000D	00013, permission denied (:retc=006Fh, EACCESS)
05	NFS protocol error	0011	00017, file/dir exists (:retc=0075h, EEXIST)
05	NFS protocol error	0012	00018, cross-device link (:retc=0090h, EXDEV)
05	NFS protocol error	0013	00019, no such device (:retc=0080h, ENODEV)
05	NFS protocol error	0014	00020, not a directory (:retc=0087h, ENOTDIR)
05	NFS protocol error	0015	00021, is a directory (:retc=0078h, EISDIR)
05	NFS protocol error	0016	00022, incorrect arguments (:retc=0079h, EIVAL)
05	NFS protocol error	001B	00027, file too large (:retc=0077h, EFBIG)
05	NFS protocol error	001C	00028, no space left on device (:retc=0085h, ENOSPC)
05	NFS protocol error	001E	00030, Read-only file system (:retc=008Dh, EROFS)
05	NFS protocol error	001F	00031, too many links (:retc=007Dh, EMLINK)
05	NFS protocol error	003F	00063, file name too long (:retc=007Eh, ENAMETOOLONG)
05	NFS protocol error	0042	00066, directory not empty (:retc=0088h, ENOTEMPTY)
05	NFS protocol error	0045	00069, disk quota exceeded (:retc=046Dh, EDQUOT)
05	NFS protocol error	0046	00070, stale file handle (:retc=046Eh, ESTALE)
05	NFS protocol error	0047	00071, too many levels of remote (:retc=046Fh, EREMOTE)
05	NFS protocol error	2711	10001, bad file descriptor (:retc=0071h, EBADF)

Table 47. Parsing error (when reason code is 6Exxyyy) (continued)

<i>xx of reasoncd</i>	Module	<i>yyyy of reasoncd</i>	Description
05	NFS protocol error	2712	10002, not sync (:retc=0071h, EBADF)
05	NFS protocol error	2713	10003, bad cookie (:retc=0076h, EFAULT)
05	NFS protocol error	2714	10004, operation not support (:retc=0086h, ENOSYS)
05	NFS protocol error	2715	10005, buffer too small (:retc=0462h, ENOBUFS)
05	NFS protocol error	2716	10006, server fault (:retc=007Ah, EIO)
05	NFS protocol error	2717	10007, bad type (:retc=008Ah, ENXIO)
05	NFS protocol error	2718	10008, jukebox (:retc=0070h, EAGAIN)
0D	UNIX System Services (USS)	0001	Abend, no SDWA (:retc=009Dh, EMVSERR)
0D	UNIX System Services (USS)	0002	Unknown Abend (:retc=009Dh, EMVSERR)
0E	UNIX System Services (USS)	<i>yyyy</i>	System Abend Code (:retc=009Dh, EMVSERR)
0F	UNIX System Services (USS)	<i>yyyy</i>	User Abend Code (:retc=009Dh, EMVSERR)
10	GFSCMAIN	<i>Inum</i>	main
11	GFSCVNAC	<i>Inum</i>	open, close, create, access, fsync, trunc, inactive
12	GFSCVNAT	<i>Inum</i>	lookup, getattr, setattr, pathconf
13	GFSCVNDR	<i>Inum</i>	mkdir, rmdir, readdir
14	GFSCVNLK	<i>Inum</i>	link, symlink, readlink
15	GFSCVNRM	<i>Inum</i>	rename, remove, recovery
16	GFSCVNRW	<i>Inum</i>	read, write
18	GFSCVMNT	<i>Inum</i>	mount, umount, statfs, pfctl, sync, recovery
19	GFSCCMNT	<i>Inum</i>	common Rnode attributes management
1A	GFSCREQB	<i>Inum</i>	reqblock/credential allocate/free
21	GFSCBIOD	<i>Inum</i>	BioD - Buffer input/output (I/O) daemon
22	GFSCMNTD	<i>Inum</i>	MntD - Mount daemon
23	GFSCBIOA	<i>Inum</i>	Common buffer management
24	GFSCBIOB	<i>Inum</i>	Buffer management
25	GFSCBIOC	<i>Inum</i>	Buffer input/output (I/O)
26	GFSCXDR2	<i>Inum</i>	XDR routines for NFS version 2 protocol
27	GFSCXDR3	<i>Inum</i>	XDR routines for NFS version 3 protocol
28	GFSC1RPC	<i>Inum</i>	Thread exit
29	GFSC0RPC	<i>Inum</i>	Thread and daemon starter
31	GFSCLOCK	<i>Inum</i>	Lock management

GFSC100E RPC REQUEST (*h_digit*) FAILED,
RETURN VALUE -1 RETURN CODE
returncd REASON CODE *rsnc* (*text*)

Explanation: Remote procedure call (RPC) request failed. (*h_digit*) is the request block address. Reason code *rsnc*, is the return code returned from TCP/IP. *text* is the failure reason. For the explanation of return code *returncd*; see *z/OS UNIX System Services Messages and Codes*.

System Action: Client continues processing.

User Response: See the return code *returncd* in the OpenEdition message manual, and reason code *rsnc* in the TCP/IP message manual.

GFSC101E NETWORK FILE SYSTEM SERVER
REQUEST FAILED (*h_digit*),
OPENEDITION RETURN CODE *returncd*
NETWORK FILE SYSTEM SERVER
RETURN CODE *retc* (*text*)

Explanation: The NFS server failed the request from the client. (*h_digit*) is the request block address. Return code *retc* is returned from NFS server. *text* is the failure reason for the request. For the explanation of return code *returncd*; see *z/OS UNIX System Services Messages and Codes*.

System Action: Client continues processing.

User Response: See the return code *returncd* in OpenEdition message manual, and the return code *retc* in the Network File System Protocol Specification, RFC 1094.

GFSC102E RPC REQUEST (*h_digit*) FAILED,
RETURN VALUE -1 RETURN CODE
returncd REASON CODE *rsnc*

Explanation: Remote procedure call (RPC) request failed. (*h_digit*) is the request block address. Reason code *rsnc*, is the return code returned from TCP/IP. The explanation of return code *returncd* is described in the OpenEdition message manual.

System Action: Client continues processing.

User Response: See the return code, *returncd*, in OpenEdition message manual, and the reason code, *rsnc*, in the TCP/IP message manual.

GFSC103E NETWORK FILE SYSTEM SERVER
REQUEST FAILED (*h_digit*),
OPENEDITION RETURN CODE *returncd*
NETWORK FILE SYSTEM SERVER
RETURN CODE *retc*

Explanation: The NFS server failed the request from the client. (*h_digit*) is the request block address. Return code *retc* is returned from the NFS server. For the

explanation of return code *returncd*; see the OpenEdition message manual.

System Action: Client continues processing.

User Response: See the return code *returncd* in OpenEdition message manual, and the return code *retc* in the Network File System Protocol Specification, RFC 1094.

GFSC105E READ FAILED, RETURN VALUE -1
RETURN CODE *returncd* REASON
CODE *reasoncd*.

Explanation: While reading a block of data from a remote file, an error *returncd* was detected.

System Action: The read operation ends. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine corrective action.

GFSC106E WRITE FAILED, RETURN VALUE -1
RETURN CODE *returncd* REASON
CODE *reasoncd*.

Explanation: While writing a block of data to a remote file, an error *returncd* was detected.

System Action: The write operation ends. NFS client processing continues. The remote file may not be complete.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine corrective action.

GFSC107E FLUSH FAILED, RETURN VALUE -1
RETURN CODE *returncd* REASON
CODE *reasoncd*.

Explanation: While flushing cached data to a remote file during close processing, an error *returncd* was detected.

System Action: The write operation ends. NFS client processing continues. The remote file may not be complete.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine corrective action.

GFSC110E *text* FAILED, RETURN VALUE -1
RETURN CODE *returncd* REASON
CODE *reasoncd*

Explanation: The NFS client has detected an error in the function, *text*.

System Action: The request has ended. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine corrective action.

System Programmer Response: Collect the detail trace log from client and from server if any.

GFSC200E VFS_MOUNT FAILED, RETURN VALUE
-1 RETURN CODE *returncd* REASON
CODE *reasoncd*.

Explanation: The **mount** command failed because of error *returncd*.

System Action: The **mount** command ended abnormally. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action, and reissue **mount** command.

GFSC201E NETWORK FILE SYSTEM CLIENT
DOES NOT SUPPORT SYNCHRONOUS
MOUNT REQUEST.

Explanation: NFS client only supports asynchronous mount.

System Action: The **mount** command ended with an error. NFS client processing continues.

User Response: Reissue **mount** command with the asynchronous option.

GFSC202E A FILE SYSTEM WITH THE SAME
NAME IS ALREADY MOUNTED.

Explanation: Cannot mount on an existing mount point.

System Action: The **mount** command ended with an error. No mount point was established. NFS client processing continues.

User Response: Reissue **mount** command with a different mount point.

GFSC203E PARSING MOUNT OPTION FAILED,
RETURN VALUE -1 RETURN CODE
returncd REASON CODE *reasoncd*
OPTION=*text*'.

Explanation: The mount option *text* was incorrectly specified.

System Action: The **mount** command ended with an error. No mount point is established. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd*, and see Table 45 on page 130 for further details of the reason code *reasoncd*. Correct the mount option, and reissue the **mount** command .

GFSC204E VFS_UMOUNT FAILED, RETURN
VALUE -1 RETURN CODE *returncd*
REASON CODE *reasoncd*.

Explanation: The **umount** or **unmount** command failed.

System Action: The **umount** or **unmount** command ended with an error. The mount point may still exist. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action. Correct it and reissue **umount** again.

GFSC205E VFS_STATFS FAILED, RETURN VALUE
-1 RETURN CODE *returncd* REASON
CODE *reasoncd*.

Explanation: The VFS_STATFS operation failed. While trying to get the status of a remote file system, an error *returncd* was detected.

System Action: The VFS_STATFS ended with an error. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action.

GFSC206E VFS_SYNC FAILED, RETURN VALUE -1
RETURN CODE *returncd* REASON
CODE *reasoncd*.

Explanation: The VFS_SYNC operation failed. While flushing cached data of remote files, an error *returncd* was detected.

System Action: The VFS_SYNC ended with an error. The remote files may not be complete. NFS client processing continues.

User Response: See *z/OS UNIX System Services*

Messages and Codes for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action.

GFSC207E VFS_RECOVER FAILED, RETURN VALUE -1 RETURN CODE *returncd* REASON CODE *reasoncd*.

Explanation: The VFS_RECOVER operation failed. While trying to recover from a previous abend, an error *returncd* was detected.

System Action: The VFS_RECOVER ended with an error. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action. If the error occurs and the program is not in error, look at the messages in the client log data sets for more information. Search problem reporting data bases for a fix for the problem. If no fix exists, contact the IBM Support Center. Provide all the printed output and copies of output data sets related to the problem.

GFSC208E VFS_PFSCTL FAILED, RETURN VALUE -1 RETURN CODE *returncd* REASON CODE *reasoncd*.

Explanation: The VFS_PFSCTL operation failed.

System Action: The VFS_PFSCTL ended with an error. NFS client processing continues.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action. If the error occurs and the program is not in error, look at the messages in the client log data sets for more information. Search problem reporting data bases for a fix for the problem. If no fix exists, contact the IBM Support Center. Provide all the printed output and copies of output data sets related to the problem.

GFSC209E THE IP ADDRESS OF REMOTE HOST NAME, *hostname*, COULD NOT BE RESOLVED.

Explanation: The VFS_MOUNT operation failed. The mount processing failed when it tried to resolve the remote host name *hostname* to a dotted IP address.

System Action: The **mount** command ended with an error. NFS client processing continues.

User Response: Correct the remote host name or use the dotted IP address of the remote host, and reissue the **mount** command.

GFSC210E NFS SERVER <hostname> DOES NOT SUPPORT NFS VERSION 3 PROTOCOL WHILE 'VERS(3) WAS SPECIFIED

Explanation: The VFS_MOUNT operation failed. The mount processing failed because the server does not support NFS version 3 protocol, which the user requested with the **mount** parameter *vers(3)*.

System Action: The **mount** command ended with an error. NFS client processing continues.

User Response: Verify that the server actually does not support NFS version 3 protocol. Remove *vers(3)* from the **mount** parameter, and reissue the **mount** command.

GFSC211E NFS SERVER <hostname> DOES NOT SUPPORT 'AUTH_UNIX' AUTHENTICATION.

Explanation: The VFS_MOUNT operation failed. The **mount** processing failed because the server does not do UNIX authentication. (AUTH_UNIX). OS/390 does not support other authentication such as Kerberos.

System Action: The **mount** command ended with an error.

User Response: Verify that the server actually does not support AUTH_UNIX authentication. Notify the server system administrator.

GFSC212E MOUNT FAILED BECAUSE OF CONVERSION SERVICE CONNECTION FAILURE CCSID *ccsid* RETC= *retc* RSNC= *rsnc*

Explanation: CCSIDs specified in the **mount** command are not supported by conversion service.

System Action: The **mount** command ended abnormally. NFS client continues processing.

User Response: See *z/OS UNIX System Services Messages and Codes* for more information about the return code *returncd* and the reason code *reasoncd* to determine corrective action

System Programmer Response: Check availability of specified CCSIDs.

GFSC213E REQUEST (*requestid*) : THE NFS SERVER *hostname* DOES NOT SUPPORT THE SPECIFIED 'VERS' AND/OR 'PROTO'.

Explanation: The **mount** request with the *requestid* failed because the user-specified *vers* or *proto* was not supported by the NFS server on the *hostname*.

System Action: The **mount** command ended with an error. NFS client continues processing.

User Response: Either let the NFS client choose the

compatible *vers* and *proto*, or determine the NFS server capabilities (by *orpcinfo*) and re-issue the **mount** command with the proper *vers* or *proto*.

GFSC214E REQUEST (*requestid*) : THE NFS SERVER *hostname* DOES NOT HAVE NFS REGISTERED ON PORT 2049

Explanation: The request with the *requestid* failed because the NFS server on the *hostname* did not register or use port 2049.

System Action: The operation ended with an error. NFS client continues processing.

User Response: Use *orpcinfo* to verify the server and correct the server.

GFSC300E MISSING LEFT PARENTHESIS IN *text* KEYWORD.

Explanation: Specified keyword, *text*, is missing a left parenthesis.

System Action: The NFS client processing stops if the error occurs in the NFS client installation parameter. The **mount** command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the **mount** parameter, correct the parameter, and reissue **mount** command. If the error occurs in the NFS client installation parameter, correct the parameter, stop OpenEdition, and restart OpenEdition.

GFSC301E PARSE FAILED ON NUMERIC FIELD FOR *text* KEYWORD.

Explanation: Specified keyword *text* contains alphabetic data in numeric field.

System Action: NFS client processing stops if the error is in the NFS client installation parameter. The **mount** command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the **mount** parameter, correct the parameter, and reissue **mount** command. If the error occurs in the NFS client installation parameter, correct the parameter, stop OpenEdition, and restart OpenEdition.

GFSC302E MISSING RIGHT PARENTHESIS IN *text* KEYWORD.

Explanation: Specified keyword *text* is missing a right parenthesis.

System Action: The NFS client processing stops if

the error is in the NFS client installation parameter. The **mount** command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the **mount** parameter, correct the parameter, and reissue **mount** command. If the error occurs in the NFS client installation parameter, correct the parameter, stop OpenEdition, and restart OpenEdition.

GFSC303E THE SPECIFIED VALUE *d_digit1* IS NOT IN THE RANGE OF *d_digit2* TO *d_digit3* FOR *text* KEYWORD.

Explanation: The value, *d_digit1*, specified in the keyword, *text*, must be between the minimum value, *d_digit2*, and the maximum value, *d_digit3*.

System Action: The NFS client processing stops if the error is in the NFS client installation parameter. The **mount** command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the **mount** parameter, correct the parameter, and reissue **mount** command. If the error occurs in the NFS client installation parameter, correct the parameter, stop OpenEdition, and restart OpenEdition.

GFSC304E PARSE FAILED ON ALPHABETIC FIELD FOR *text* KEYWORD.

Explanation: Specified keyword, *text*, contains numeric data for alphabetic field.

System Action: The NFS client processing stops if the error is in the NFS client installation parameter. The **mount** command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the **mount** parameter, correct the parameter, and reissue **mount** command. If the error occurs in the NFS client installation parameter, correct the parameter, stop OpenEdition, and restart OpenEdition.

GFSC305E INCORRECT OPTION *text1* SPECIFIED FOR *text2* KEYWORD, VALID OPTION IS Y OR N.

Explanation: An incorrect option, *text1*, was specified for the keyword, *text2*.

System Action: The NFS client processing stops if the error is in the NFS client installation parameter. The

mount command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the **mount** parameter, correct the option for the keyword *text2* and reissue **mount** command. If the error occurs in the NFS client installation parameter, correct the option for the keyword, *text2*, stop OpenEdition, and restart OpenEdition.

GFSC307E *text* IS AN INCORRECT KEYWORD FOR MOUNT PARAMETER.

Explanation: *text* can only be specified as an installation parameter.

System Action: The **mount** command failed.

System Programmer Response: Correct the **mount** parameter keyword *text*.

GFSC308E REQUEST (*requestid*) : XLAT(Y) CANNOT BE SPECIFIED AS A MOUNT PARAMETER WHEN TAG OPTION IS ALSO SPECIFIED.

Explanation: The **mount** request with the *requestid* code failed because the user specified both *xlat(Y)* and *tag* options together. This is not allowed.

System Action: The **mount** command ended with an error. NFS client continues processing.

User Response: Either specify *xlat(Y)* to have the NFS client do text translation based on the *cln_ccsid* and *srv_ccsid* values, or specify *tag* option with the proper Coded Character Set Identifier (CCSID) to have the translation done by LFS based on the CCSID in the *tag* option.

GFSC309E UNKNOWN KEYWORD ENCOUNTERED AROUND POSITION *d_digit*.

Explanation: The keyword specified in position *d_digit* is not a valid keyword.

System Action: The NFS client processing stops if the error is in the NFS client installation parameter. The **mount** command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the **mount** parameter, correct the keyword, and reissue the **mount** command. If the error occurs in the NFS client installation parameter, correct the keyword, stop OpenEdition, and restart OpenEdition.

GFSC310I READAHEAD AND DELAYWRITE OPTIONS WILL BE IGNORED AS DATACACHING IS OFF.

Explanation: The keywords *ReadAhead* and *DelayWrite* will be ignored because **DataCaching** has been set OFF.

System Action: NFS client continues processing.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Check parameters to make sure that **DataCaching** should be OFF.

GFSC311I CLN_CCSID AND SRV_CCSID WILL BE IGNORED AS XLAT OPTION IS OFF.

Explanation: The keywords *cln_ccsid* and *srv_ccsid* will be ignored because **xlat** has been set off.

System Action: NFS client continues processing.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Check parameters to make sure that **xlat** should be off.

GFSC312I ACREGMIN, ACREGMAX, ACDIRMIN, AND ACDIRMAX OPTIONS WILL BE IGNORED AS ATTRCACHING IS OFF.

Explanation: The keywords *acregmin*, *acregmax*, *acdirmin*, and *acdirmax* will be ignored because **AttrCaching** has been set OFF.

System Action: NFS client continues processing.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Check parameters to make sure that **AttrCaching** should be OFF.

GFSC313I RETRANS OPTION WILL BE IGNORED AS HARD OPTION IS ON.

Explanation: The keyword *retrans* will be ignored because *hard* has been set ON.

System Action: NFS client continues processing.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Check parameters to make sure that *hard* should be ON.

GFSC315E ERROR ENCOUNTERED WHILE PARSING MOUNT PATH, REASON CODE *reasoncd*.

Explanation: The specified mount path is not correct.

System Action: The **mount** command failed.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: See *z/OS UNIX System Services Messages and Codes* for a description of the reason code *reasoncd*. Correct the mount path, and reissue mount command.

GFSC317E ERROR ENCOUNTERED WHILE PARSING HOSTNAME, REASON CODE *reasoncd*.

Explanation: The specified hostname is not correct.

System Action: The mount command failed.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: See *z/OS UNIX System Services Messages and Codes* for a description of the reason code *reasoncd*. Correct the mount path, and reissue mount command.

GFSC318E READ FAILED FOR NETWORK FILE SYSTEM CLIENT MOUNT PARAMETERS.

Explanation: An error occurred while the NFS client was processing the **mount** parameters. This message follows other messages, GFSC3xxE, that describe the error in greater detail.

System Action: The mount command failed.

System Programmer Response: Correct the **mount** parameter options, and reissue **mount** command.

GFSC319E *text* IS AN INCORRECT KEYWORD FOR NETWORK FILE SYSTEM CLIENT INSTALLATION PARAMETER.

Explanation: *text* can only be specified as a mount parameter.

System Action: NFS client processing stops.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Correct the NFS client installation parameter, stop OpenEdition, and restart OpenEdition.

GFSC320E INCORRECT OPTION *text* SPECIFIED FOR DELIM KEYWORD, VALID OPTION IS BINARY, CR, CRLF, CRNL, LF, LFCR, OR NL.

Explanation: An incorrect option *text*, has been specified for the keyword *delim*.

System Action: NFS client processing stops if the error is in the NFS client installation parameter. The

mount command failed if the error is in the **mount** parameter.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: If the error is in the mount parameter, correct the option for the keyword *delim*, and reissue mount command. If the error occurs in the NFS client installation parameter, correct the option for the keyword *delim*, stop OpenEdition, and restart OpenEdition.

GFSC500I CLIENT LOG DATA SET, *text*, FLUSHED.

Explanation: The data buffer of the active client log data set *text*, was flushed to disk. *text* is the associated data set name of the active client log data set.

System Action: NFS client processing continues.

GFSC501I CLIENT LOG DATA SET *text* RE-INITIALIZED.

Explanation: The error log data set is re-initialized. *text* is the associated data set name of the active log data set.

System Action: NFS client processing continues.

GFSC502E CANNOT OPEN CLIENT LOG DATA SET, *text1*, *text2*.

Explanation: NFS client failed to open client log data set. *text1* is the DD associated with the client log data set which cannot be opened. *text2* is the failure reason of the *C* function, *fopen*.

System Action: NFS client processing stops.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Fix client log data set and reply to the OpenEdition message to restart the NFS client.

GFSC503E CLIENT LOGGING ENDED.

Explanation: NFS client failed to manipulate the client log data set. See previous operator console message for the failure reason.

System Action: NFS client processing continues.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Fix client log data set, stop the NFS client, and reply to the OpenEdition message to restart the NFS client.

GFSC504I CLIENT LOG DATA SET SWITCHED TO
text.

Explanation: A "no space" or an I/O error condition was detected while writing to the client log data set. NFS client logging switched to the other log data set, *text*.

System Action: NFS client processing continues.

**GFSC505E MISSING DD STATEMENT OR
INCORRECT DATA SET
ORGANIZATION FOR LOG DATA SET.**

Explanation: The error log data set has incorrect data set organization, or missing DD statement.

System Action: NFS client processing stops if the error occurred during initialization time. NFS client processing continues with client logging ended if the error occurred after initialization time.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Correct error log data set DD statement or data set organization, stop the NFS client, and reply to the OpenEdition message to restart the NFS client.

GFSC506E *text1* FAILED FOR *text2*, *text3*.

Explanation: NFS client failed to manipulate the client log data set, *text2*. *text3* is the failure reason for the C function, *text1*.

System Action: NFS client processing continues.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Fix client log data set, stop the NFS client, and reply to the OpenEdition message to restart the NFS client.

**GFSC700I OS/390 *version* NETWORK FILE
SYSTEM CLIENT *fmid* STARTED**

Explanation: The NFS client is initialized and ready to process NFS requests.

In the message text:

version The version number, for example:
 VERSION 1.

fmid The NFS client FMID started, for example:
 (HDZ11TC).

System Action: NFS client continues processing.

**GFSC701I NETWORK FILE SYSTEM CLIENT
SHUTDOWN IN PROGRESS.**

Explanation: NFS client shutdown processing has started.

System Action: NFS client shutdown processing continues.

**GFSC702I NETWORK FILE SYSTEM CLIENT
SHUTDOWN COMPLETE.**

Explanation: NFS client has completed shutdown processing.

System Action: NFS and its associated subtasks have ended.

**GFSC703E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: NETWORK
FILE SYSTEM CLIENT IS ALREADY
STARTED.**

Explanation: Only one NFS client can be started on an MVS system.

System Action: This NFS client ends.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: A prior NFS client session has not ended. UNIX end processing should have ended the NFS client colony address space. Collect installation parameters, dumps, NFS client log data sets, and contact IBM programming support personnel. Use the force command to end the NFS client colony address space and then restart UNIX.

**GFSC704E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: DOWN LEVEL
SECURITY PRODUCT.**

Explanation: Resource Access Control Facility (RACF) MVS security product is down level.

System Action: The NFS client ends.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Check to determine the required RACF level.

**GFSC705E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: MAIN
STORAGE IS UNAVAILABLE.**

Explanation: The NFS client was not able to allocate the necessary storage. The cause might be the value specified on the *bufhigh* attribute is too large or the REGION size is too small.

System Action: The NFS client ends.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: The requested memory is not available during the NFS client initialization. Do one or both of the following before restarting the NFS client:

- Increase the REGION size for the client procedure.
- Decrease the value specified for the bufhhigh attribute of the FILESYSTYPE parameter in the BPXPRMxx Parmlib member.

**GFSC707E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: INCORRECT
PARAMETER IN INSTALLATION
PARAMETERS.**

Explanation: The NFS client has detected an error in the installation parameters.

System Action: The NFS client ends.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: Check previous console messages prefixed with GFSC. Correct the parameter, stop OpenEdition, and restart OpenEdition.

**GFSC708E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: CONVERSION
SERVICE IS NOT INSTALLED OR NOT
AVAILABLE. RETURN CODE *retc*,
REASON CODE *rsnc*.**

Explanation: Both Unicode and Character Data Representation Architecture (CDRA) initialization requests failed.

System Action: The NFS client startup ends.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: See *z/OS MVS System Messages, Vol 2 (ASB-EWX)* for a description of the return code *retc* and the reason code *rsnc* to determine the corrective action.

**GFSC709E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: NOT
STARTED IN A STANDALONE COLONY
ADDRESS SPACE.**

Explanation: The NFS client has detected an error during initialization processing. the NFS client was started by some other means, other than UNIX kernel.

System Action: The NFS client ends.

Operator Response: Record MVS operator console message and notify the MVS system programmer.

System Programmer Response: The NFS client must be initialized by UNIX kernel.

**GFSC710E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: ESTAE
INITIALIZATION FAILED.**

Explanation: The NFS client has detected an error during the initialization processing.

System Action: The NFS client ends.

Operator Response: Collect any dumps, NFS client log data sets, and notify the system programmer.

System Programmer Response: Collect installation parameters, dumps, NFS client log data sets, and Contact IBM programming support personnel.

**GFSC711E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: OPENEDITION
KERNEL SERVICE FAILED.**

Explanation: The NFS client has detected an error during the second phase of initialization processing.

System Action: The NFS client ends.

Operator Response: Collect any dumps, NFS client log data sets, and notify the system programmer.

System Programmer Response: Collect installation parameters, dumps, NFS client log data sets, and contact IBM programming support personnel.

**GFSC712E NETWORK FILE SYSTEM CLIENT
INITIALIZATION FAILED: SOCKET
CALL GETHOSTNAME FAILED,
RETURN CODE *returncd*.**

Explanation: The NFS client has detected an error (*returncd*) during the initialization processing. This error may be caused by OpenEdition and TCP/IP connection failure.

System Action: The NFS client ends.

Operator Response: Collect any dumps, NFS client log data sets, and notify the system programmer.

System Programmer Response: Collect installation parameters, dumps, NFS client log data sets, and Contact IBM programming support personnel.

**GFSC713E NETWORK FILE SYSTEM CLIENT
LOGGING IS TERMINATED.**

Explanation: The NFS client has detected an error. The Client Log daemon has ended.

System Action: The NFS client ends.

Operator Response: Collect any dumps, NFS client log data sets, and notify the system programmer.

System Programmer Response: Collect installation parameters, dumps, NFS client log data sets, and Contact IBM programming support personnel.

**GFSC714E NETWORK FILE SYSTEM CLIENT SVC
DUMP REQUEST FAILED, RETURN
CODE *returncd* REASON CODE
reasoncd.**

Explanation: A request to write an MVS SVC dump failed. See the description of the **sdump** macro in *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU* for the meaning of the return code *returncd* and reason code *reasoncd*.

System Action: The NFS client ends.

Operator Response: Collect the MVS console log, NFS client log data sets, and notify the system programmer.

System Programmer Response: Collect installation parameters, dumps, NFS client log data sets, and Contact IBM programming support personnel.

**GFSC715E NETWORK FILE SYSTEM CLIENT
ESTAE EXIT UNABLE TO COMPLETE,
ABEND CODE *d_digit*.**

Explanation: The NFS client recovery exit has detected a recursive abend. The *d_digit* is the last abend code encountered by a secondary instance of the ESTAE exit routine.

System Action: The ESTAE processing stops. The NFS client ends.

Operator Response: Collect any dumps, the MVS console log, NFS client log data sets, and notify the system programmer.

System Programmer Response: Collect installation parameters, dumps, the MVS console log, NFS client log data sets, and Contact IBM programming support personnel.

**GFSC716I NETWORK FILE SYSTEM CLIENT
DAEMON TASK RESTARTED.**

Explanation: The NFS client detected a stopped asynchronous daemon task. The NFS client has restarted the stopped asynchronous daemon task.

System Action: The NFS client daemon has restarted. NFS client processing continues.

**GFSC721E UNABLE TO SETUP ERROR
RECOVERY (ESTAE), RETURN CODE
returncd.**

Explanation: NFS client daemon or thread failed to setup error recovery. See *z/OS MVS Programming: Authorized Assembler Services Reference ENF-IXG* for information the **estae** macro return code, *returncd*.

System Action: If the error occurred in the daemon, the NFS client has initiated shutdown processing. If the error occurred in the thread, the associated operation

ends and NFS client continues processing.

Operator Response: Collect any dumps, the MVS console log, NFS client log data sets, and notify the system programmer.

System Programmer Response: If the error occurs and the program is not in error, look at the messages in the client log data sets for more information. Search problem reporting data bases for a fix for the problem. If no fix exists, contact the IBM Support Center. Provide all the printed output and copies of output data sets related to the problem.

**GFSC722E A SOCKET COULD NOT BE CREATED,
RETURN VALUE -1 RETURN CODE
returncd REASON CODE *reasoncd*.**

Explanation: NFS client Daemon or thread processing failed to create a socket for network communication.

System Action: If the error occurred in the daemon, the NFS client has initiated shutdown processing. If the error occurred in the thread, the associated operation ends and the NFS client continues processing.

Operator Response: Collect any dumps, the MVS console log, NFS client log data sets, and notify the system programmer.

System Programmer Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action. If the error occurs and the program is not in error, look at the messages in the client log data sets for more information. Search problem reporting data bases for a fix for the problem. If no fix exists, contact the IBM Support Center. Provide all the printed output and copies of output data sets related to the problem.

**GFSC723E NETWORK FILE SYSTEM CLIENT
ABEND @ *module+offset*.**

Explanation: NFS client encountered a programming error. SVC dump was issued to capture the diagnostic information.

System Action: NFS client has initiated shutdown processing.

System Programmer Response: Look at the messages in the client log data sets for more information. Search problem reporting data bases for a fix for the problem. If no fix exists, contact the IBM Support Center. Provide all the printed output and copies of output data sets related to the problem.

**GFSC724E UNABLE TO BIND A RESERVED PORT
TO SOCKET *socketnum*.**

Explanation: The mount daemon or umount thread failed to obtain a reserved port for socket, *socketnum*, for network communication.

System Action: If the error occurred in the daemon, the NFS client has initiated shutdown processing. If the error occurred in the thread, the associated operation ends and the NFS client continues processing.

System Programmer Response: If the error occurs and the program is not in error, look at the messages in the client log data sets for more information. Search problem reporting data bases for a fix for the problem. If no fix exists, contact the IBM Support Center. Provide all the printed output and copies of output data sets related to the problem.

GFSC725E SOCKET, *socketnum*, COULD NOT BE CLOSED, RETURN VALUE -1 RETURN CODE *returncd* REASON CODE *reasoncd*.

Explanation: While closing a socket *socketnum*, an error *returncd* was detected.

System Action: NFS client continues processing.

System Programmer Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return code *returncd* and the reason code *reasoncd* to determine the corrective action. If the error occurs and the program is not in error, look at the messages in the client log data sets for more information. Search problem reporting data bases for a fix for the problem. If no fix exists, contact the IBM Support Center. Provide all the printed output and copies of output data sets related to the problem.

GFSC726E REQUEST (*requestid*) : THE PORTMAPPER OF THE SERVER *hostname* DOES NOT RESPOND TO PMAP_GETMAPS

Explanation: The **mount** request with the *requestid* code failed because the user specified *proto(tcp)* and the *hostname* did not publish its registered remote program port for remote procedure call (RPC).

System Action: The **mount** command ended with an error. NFS client continues processing.

User Response: Verify that the server does not respond to *orpcinfo*. Re-issue the **mount** command without *proto(tcp)*.

GFSC727W REQUEST (*requestid*) : THE PORTMAPPER OF THE SERVER *hostname* DOES NOT RESPOND TO PMAP_GETMAPS. ATTEMPTING UDP RPC WITH NFS VERSION *d_digit* PROTOCOL.

Explanation: The NFS client warns the user who mounts to the *hostname* that the host may not have its **Portmapper** running; the NFS client uses user datagram protocol (UDP) remote procedure call (RPC). A user datagram protocol (UDP) remote procedure call

(RPC) is attempted with NFS protocol.

System Action: The **mount** command may or may not succeed. NFS client continues processing.

User Response: Verify that the server does not respond to the *orpcinfo* code. If the **mount** command subsequently fails, then it is likely that the host does not have the NFS server.

GFSC728E REQUEST (*requestid*) : UNABLE TO CONNECT SOCKET=*socketno* PORT=*portno1* WITH THE SERVER *hostname* PORT=*portno2*

Explanation: The NFS client uses TCP/IP to connect the socket *socketno* with port *portno1* to the server *hostname* with port *portno2*. However, the connect system call failed.

System Action: The operation ended with an error. The NFS client continues processing.

User Response: Verify that the NFS server on *hostname* is available and running. Verify that the TCP/IP subsystem on MVS is available and running. If the **mount** command subsequently fails, then it is very likely that the host does not have the NFS server.

GFSC729W OS/390 NFS CLIENT IS UNABLE TO RESERVE *d_digit* SOCKETS, CURRENT MAXFILEPROC IS *d_digit*

Explanation: The maximum number of socket descriptors that a process can have open concurrently has been exceeded. Increase the MAXFILEPROC parameter value in the BPXPRMxx member to bypass this problem. See *z/OS UNIX System Services Planning* for more information about specifying the MAXFILEPROC value.

System Action: The operation ended with an error. The NFS client continues processing.

System Programmer Response: Increase the MAXFILEPROC parameter value in the BPXPRMxx member to bypass the problem.

GFSC840I usage: *text* [-a] [-d] [-e] [*host*]

Explanation: This is the usage for the **showmount** command. *text* is the command as entered by the user. The valid options are as follows:

- a Display all mounts in the format
Hostname:Directory from **host** NFS server
- d Display only directory names of all mounts from **host** NFS server
- e Display the list of exported directories from **host** NFS server

GFSC841E Unknown host *text*

Explanation: The user entered incorrect host address information, *text*.

System Action: Command stops processing.

User Response: Correct syntax and re-issue the command.

GFSC842E Cannot resolve local host name

Explanation: Local host name is not found.

System Action: Command stops processing.

User Response: Contact your system administrator to check TCP/IP configuration.

GFSC843E Unknown flag '-character'

Explanation: An incorrect option, '-character', is specified.

System Action: Command stops processing.

User Response: Correct syntax and re-issue the command.

GFSC845I usage: *text* input output

Explanation: This is the usage for the **os22mvs** and **mvs2os2** commands. *text* is the command as entered by the user. Table 48 shows the valid parameters:

Table 48. GFSC845I valid parameters

<i>input</i>	Absolute path name of the input file to be converted.
<i>output</i>	Absolute path name of the output file.

GFSC846E Cannot open input file, *text1:text2*

Explanation: Cannot open input file, *text1*. *text1* is the input path name as entered by the user. *text2* is the failure information returned when attempting to open the input file.

System Action: Command stops processing.

User Response: Check input file, *text1*.

GFSC847E Cannot open output file, *text1:text2*

Explanation: Cannot open output file, *text1*. *text1* is the output path name as entered by the user. *text2* is the failure information returned when attempting to open the output file.

System Action: Command stops processing.

User Response: Check output file, *text1*.

GFSC848E Cannot read input file, *text1: text2*

Explanation: Cannot read input file, *text1*. *text1* is the input path name as entered by the user. *text2* is the failure information returned when attempting to read the input file.

System Action: Command stops processing.

User Response: Check input file, *text1*.

GFSC849E Cannot write output file, *text1: text2*

Explanation: Cannot write output file, *text1*. *text1* is the output path name as entered by the user. *text2* is the failure information returned when attempting to write the output file.

System Action: Command stops processing.

User Response: Check output file, *text1*.

GFSC850E Input path name cannot be equal to output path name.

Explanation: Input path name cannot be equal to output path name.

System Action: Command stops processing.

User Response: Correct syntax and re-issue the command.

GFSC854I usage: *text* [-crnzm <mount point>]

Explanation: This is the usage for the **nfsstat** command. *text* is the command as entered by the user. The valid parameters are as follows:

- c Display both NFS and remote procedure call (RPC) statistics about the NFS client
 - n Display NFS statistics about the NFS client
 - r Display remote procedure call (RPC) statistics about the NFS client
 - z Initializes statistics to zero. This is for use by root user only and can be combined with any of the above options. Zero particular set of statistics after printing them.
 - m Display the name of each NFS mounted file system
 - m *mount point*
Display information of NFS mounted file system on the specified mount point
-

GFSC855E Must be a root user to issue '-character' flag

Explanation: The option, '-character', can only be issued with the root authority.

System Action: Command stops processing.

User Response: Contact your system administrator to issue this command.

GFSC856E Network File System Client command, *text*, failed, return value -1 return code *returncd* reason code *reasoncd*

Explanation: The command, *text*, failed.

System Action: Command stops processing.

User Response: See *z/OS UNIX System Services Messages and Codes* for a description of the return

code, *returncd*. See Table 45 on page 130 for more information about the reason code, *reasoncd*.

GFSC858E Directory *text* not mounted.

Explanation: The directory, *text*, was not mounted.

System Action: Command stops processing.

User Response: Issue `nfsstat -m` to view the list of active mount points. If the mount point does not exist, contact system administrator to mount the directory.

Messages from the client platform (AIX)

This section provides a list of messages from the client operating system, in response to NFS reply results (messages without message numbers). These messages are platform dependent.

Note: The message text is from an AIX RS/6000.

This section contains messages from the client's operating system.

Cross device link

Explanation:

1. An attempt has been made to rename a member of a PDS or PDSE, but the target file is not a member of the same PDS or PDSE.
2. An attempt has been made to rename a non-PDS or PDSE file, but the target file is a member of a PDS or PDSE.

User Response: Try a copy and remove instead of a rename.

Directory Not Empty

Explanation: An attempt has been made to remove a PDS or PDSE that has members.

User Response: Delete members before trying to remove a PDS or PDSE.

File exists

Explanation: An attempt has been made to rename a PDS or PDSE, but the target file already exists.

User Response: Delete the target file before renaming a PDS or PDSE. This is not required for a regular file or for a PDS or PDSE member.

File Name Too Long

Explanation: The name is not a valid MVS file or member name.

User Response: File names must follow the MVS naming conventions. See *z/OS DFSMS: Using Data Sets* for MVS file naming conventions.

Invalid

Explanation:

1. The specified parameters were incorrect.
2. The creation of a VSAM data set failed.

User Response: Respecify the correct parameters or contact the system programmer to determine the VSAM data set failure on the NFS server.

I/O Error (with possible system programmer response)

Explanation:

1. Unexpected error from Catalog Management.
2. Dynamic Allocation failed during action other than a read or write.
3. A file could not be opened during an action other than a read or write.
4. An error occurred while reading a partitioned data set (PDS) or PDSE directory.
5. No space is available in task input/output table (TIOT).
6. TIOT resource is unavailable.
7. Unable to release enough resources.
8. Insufficient units are available.
9. The server could not get enough memory to perform this function.

System Programmer Response: For (9), stop the server and change the region field in the job control language (JCL) before restarting, or modify parameters in the attributes file. For the other possible explanations, perform the appropriate action.

I/O Error (with possible user response)

Explanation:

1. An attempt has been made to nest PDSs or PDSEs.
2. Maximum number of file allocations is exceeded.
3. The file is being written in text mode, the new write request offset is determined to fall within the end-of-line (EOL) sequence (lf, cr, crlf, lfcr) of a previous line, and the new data does not contain the correct EOL characters.
4. The file is being written in text mode, with a non-zero EOL (lf, cr, lfcr, crlf). The number of bytes of data in the written line is larger than the maximum record size of the file.
5. The file is being written in text mode, with fixed records, with a non-zero EOL, **blankstrip** not set (no padding blanks on write), and the number of bytes of data in the written line is less than or greater than the record size of the file.
6. The file is being written in text mode, with fixed records, **blankstrip** set, and the line of written data contains trailing blanks as part of the data.
7. When a workstation file containing a 0 length line is written to MVS as recfm(u) in text mode, a write error occurs.
8. An MVS Access Method Services alias name was specified in a remove (**rm** or **rmdir**) or rename (**mv**) request.
9. An "s" was specified in the recfm attribute for a PDS or PDSE.
10. If you try to append data to a member of a PDS or PDSE, an I/O error occurs.
11. An incorrect attribute was specified in the command.

User Response: Perform the appropriate action.

Is a directory

Explanation: A non-directory operation has been tried on a PDS or PDSE.

User Response: Use directory operations on the file.

Network File System server *name* not responding still trying

Explanation: Long delays between operations.

User Response: The user response is:

- The server might need extra time to service client requests. Wait until the message "Network File System server *name* ok" appears.
- Determine if the network traffic is heavy and overloaded. Try to isolate the path where the client workstation communicates with the server machine.

- Determine if the client workstation transmits the same requests over and over. Commands such as **nfsstat -c** on AIX or UNIX platforms show the number of client retransmissions (**retrans**) as well as the number of **badcalls** and **badxid**. When the number of **badcalls** and **badxid** are high, the client machine usually has bad retransmissions. High retransmissions might be caused by an overloaded network or a slow server.
- If the network is overloaded, contact your network administrator.
- If the server is slow, determine if your client workstation tends to transmit requests out-of-sequence or incompletely, and you are performing I/O to a file in text mode. If the request is incomplete, the client may not send the remainder of the request until a later time. In this case, increase the server's **cachewindow** attribute. The **cachewindow** buffers store out-of-sequence and incomplete requests from clients. The general rule in setting the *n* value of **cachewindow(n)** is $n = ((\text{num of BIOD} + 1) * (\text{client_max_IO_buffer_size}/\text{transfer_size}))$ where,
 - BIOD is the number of blocked I/O daemons set by the client workstation. This value is usually set to defaults at the installation of the operating system or by your system administrator.
 - **client_max_IO_buffer_size** is the amount of I/O data that is requested by the client (for example, the client writes 8192 bytes of data to the remote file system). This value is determined by your application programs.
 - **transfer_size** is the actual size of data being sent across the network (for example, the 8192 bytes of data may be broken down to 16 smaller packets of 512 bytes (16x512=8192)). This value is determined dynamically by your client workstation.
- If your client workstation tends to send duplicate transmissions for the same request too often, thus increasing the workload on the server, you might want to delay the client's retransmission rate and request's timeout. On the mount command, you can specify: `mount -o retrans=3,timeout=30 IO_buffer_size`, where:

retrans Is the number of retransmissions allowed before a timeout. Default is vendor-specific ranging from **3-5**.

timeout Is the timeout value in tenths of a second. `timeout=30` means 3 seconds. Default is vendor specific ranging from **5-11**.

If the reply is not received by the client within the timeout period, a minor timeout has occurred for this request. The timeout period is doubled, and the

request is sent again. The process is repeated until the retransmission count specified by the **retrans** is reached; if no reply has been received, major timeout has occurred.

No such device**Explanation:**

1. The file resides on an off-line device.
2. The file has been migrated to another storage level. Whether it is being recalled depends on the **retrieve** attribute.

User Response: For (1), contact your MVS operator or MVS system programmer. For (2), try the request again later, if the **retrieve** attribute is enabled. If not, try the request again, with the **retrieve** attribute enabled.

No space left on device**Explanation:**

1. The file has exceeded the space that was allocated to it.
2. The PDS or PDSE has exceeded the space that was allocated to it.
3. The PDS directory has exceeded the space that was allocated to it.

User Response:

1. For (1), save this file into a larger file and then rename it to the old name, if necessary.
2. For (2), create a larger PDS or PDSE and store this member there. Then copy the members of the old PDS or PDSE to the new PDS or PDSE. Rename the new PDS or PDSE if necessary.
3. For (3), Create a new PDS with a larger directory (use the **dir** attribute). Store this member in the new PDS. Then copy the members from the old PDS to the new PDS and rename them, if necessary.

No such file or directory

Explanation: A **locate** command failed for this file. The file is not cataloged, or the MVS system operator might have unmounted the file before you issued the **umount** command.

User Response: Check your spelling. If it is correct, contact your system administrator.

Not a directory

Explanation: A directory operation has been tried on a file that is not a PDS or PDSE.

User Response: Use non-directory operations on the file.

Not Owner**Explanation:**

1. File has not timed out yet.
2. File is open. Another client (maybe even the same client) has the file open for writing.
3. The client tried to change the mode in a **nfsattr** Network File System procedure call.

User Response: The user response is:

- For (1), follow the steps in waiting and retrying that are described under the "Permission denied" message.
- For (3), do not try to change the mode.

Permission denied**Explanation:**

1. The file is not in use, but has not timed out yet (occurs most often when writing PDS or PDSE members without **writetimeout** seconds expiring between saving members).
2. The file is in use by an MVS user, or another client.
3. Dynamic allocation—Authorized function requested by an unauthorized user. The error codes from dynamic allocation are printed to the log data set.
4. Resource access control facility (RACF) is not active.
5. Dynamic allocation—Request denied by operator (dsname allocation). The error codes from dynamic allocation are printed to the log data set.
6. Dynamic allocation—Installation validation routine denied this request. The error codes from dynamic allocation are printed to the log data set.
7. You are not authorized for this request.
8. The MVS operator suspended mount request processing (only if this message appears following a mount attempt).
9. The file/prefix is not exported to your client (only if this message appears following a mount attempt).
10. IDCAMS failed during a rename or remove procedure. Usually this happens because the file is in use. The output from IDCAMS is printed to the log data set.
11. With OS/2, you might get a "SYS0055 Access denied" message if the **noretrieve** attribute is set and a **dir** command is done against a mounted file system containing migrated files.

User Response: For (1), To determine if this is the problem, check the timeout values (**attrtimeout**, **readtimeout**, **writetimeout**). Retry the request after the shortest timeout has expired. If the request still fails, retry the request after the next shortest timeout has expired. If it still fails, retry after the longest timeout has expired. If the request still fails, this is not the problem. For (2) and (10), try the request again later. For (3), (4),

(5), (6), and (9), notify your MVS system programmer. For (7), enter the **mvslogin** command again and retry the request. If the request still fails, notify your MVS system programmer. For (8), notify your MVS operator or MVS system programmer. For (11), specify the **retrieve** attribute on the **mount** command, or the MVS system administrator can make that the default.

Read Only File System

Explanation: One of these Network File System procedures was tried on a read-only file system: link, write, rename, remove, mkdir, or create.

User Response: See the documentation on the exports data set to see how a file system is designated read-only (see *z/OS Network File System Customization and Operation*). The exports data set needs to be changed, or you are using it incorrectly.

Stale NFS File Handle

Explanation: A file handle is used by the client and server sides of the Network File System to specify a particular file or prefix. A stale file handle occurs when the name is no longer valid, possibly due to one of the these conditions:

1. The file or prefix has been removed by the MVS operator.
2. The server has been stopped and brought back up. This affects files and members below mount points.

User Response: For (1), unmount and mount again. If your client maintains that the device is busy even though it is not, you might have to restart your client. For (2), enter the **mvslogin** command again and retry the request.

Weak Authorization

Explanation: The authorization data in the remote procedure call (RPC) message was not valid. This is a client side error.

User Response: UNIX-style authorization is required.

Appendix B. Return codes

Table 49 lists the externalized return codes that are defined by the NFS Version 2 protocol.

Table 49. Externalized return codes defined by the NFS version 2 protocol

Return Value	Return Code	Description
NFS_OK	0	Requests completed successfully and the results are valid.
NFSERR_PERM	1	Not owner. The caller does not have correct ownership to perform the requested operation.
NFSERR_NOENT	2	No such file or directory. The file or directory specified does not exist.
NFSERR_IO	5	A hard error occurred when the operation was in progress. For example, this could be a disk error.
NFSERR_NXIO	6	No such device or address.
NFSERR_ACCESS	13	Permission denied. The caller does not have the correct permission to perform the requested operation.
NFSERR_EXIST	17	File exists. The file specified already exists.
NFSERR_NODEV	19	No such device.
NFSERR_NOTDIR	20	Not a directory. The caller specified a non-directory in a directory operation.
NFSERR_ISDIR	21	Is a directory. The caller specified a directory in a non-directory operation.
NFSERR_EINVAL	22	An argument was passed to the z/OS NFS server was not valid.
NFSERR_FBIG	27	File too large. The operation caused a file to grow beyond the server's limit.
NFSERR_NOSPC	28	No space left on device. The operation caused the server's file system to reach its limit.
NFSERR_ROFS	30	Read-only file system. Write tried on a read-only file system.
NFSERR_NAMETOOLONG	63	File name too long. The file name in an operation was too long.
NFSERR_NOTEMPTY	66	Directory not empty. Tried to remove a directory that was not empty.
NFSERR_DQUOT	69	Disk quota exceeded. The client's disk quota on the server has been exceeded.
NFSERR_STALE	70	The file handle given in the arguments was not valid. That is, the file referred to by that file handle no longer exists, or access to it has been revoked.

Table 50 lists the externalized return codes that are defined by the NFS version 3 Protocol.

Table 50. Externalized return codes defined by the NFS version 3 protocol

Return Value	Return Code	Description
NFS_OK	0	Requests completed successfully and the results are valid.

Table 50. Externalized return codes defined by the NFS version 3 protocol (continued)

Return Value	Return Code	Description
NFS3ERR_PERM	1	Not owner. The caller does not have correct ownership to perform the requested operation.
NFS3ERR_NOENT	2	No such file or directory. The file or directory specified does not exist.
NFS3ERR_IO	5	A hard error occurred when the operation was in progress. For example, this could be a disk error.
NFS3ERR_NXIO	6	No such device or address.
NFS3ERR_ACCESS	13	Permission denied. The caller does not have the correct permission to perform the requested operation.
NFS3ERR_EXIST	17	File exists. The file specified already exists.
NFS3ERR_XDEV	18	Attempt to do an operation across the file system.
NFS3ERR_NODEV	19	No such device.
NFS3ERR_NOTDIR	20	Not a directory. The caller specified a non-directory in a directory operation.
NFS3ERR_ISDIR	21	Is a directory. The caller specified a directory in a non-directory operation.
NFS3ERR_INVAL	22	An argument was passed to the z/OS NFS server was not valid.
NFS3ERR_FBIG	27	File too large. The operation caused a file to grow beyond the server's limit.
NFS3ERR_NOSPC	28	No space left on device. The operation caused the server's file system to reach its limit.
NFS3ERR_ROFS	30	Read-only file system. Write tried on a read-only file system.
NFS3ERR_MLINK	31	Too many links.
NFS3ERR_NAMETOOLONG	63	File name too long. The file name in an operation was too long.
NFS3ERR_NOTEMPTY	66	Directory not empty. Tried to remove a directory that was not empty.
NFS3ERR_DQUOT	69	Disk quota exceeded. The client's disk quota on the server has been exceeded.
NFS3ERR_STALE	70	The file handle given in the arguments was not valid. That is, the file referred to by that file handle no longer exists, or access to it has been revoked.
NFS3ERR_NOT_SYNC	10001	File handle is not valid.
NFS3ERR_BAD_COOKIE	10002	Synchronization mismatch on SETATTR.
NFS3ERR_BAD_COOKIE	10003	REaddir and REaddirPlus cookie is stale.
NFS3ERR_NOTSUPP	10004	Operation is not supported.
NFS3ERR_TOOSMALL	10005	Buffer or request is too small.
NFS3ERR_SERVERFAULT	10006	Server abandons the request.
NFS3ERR_BADTYPE	10007	Type of an object is not supported.
NFS3ERR_JUKEBOX	10008	Request was initiated, but not completed.

Table 51 on page 151 lists the externalized return codes that are defined by the NFS protocol that is mapped to the z/OS UNIX return codes.

Table 51. z/OS NFS version 1 protocol externalized return code mapping to z/OS UNIX

NFS Protocol Value	Return Code	z/OS UNIX Value	Return Code
NFSERR_PERM	1	EPERM	139
NFSERR_NOENT	2	ENOENT	129
NFSERR_IO	5	EIO	122
NFSERR_NXIO	6	ENXIO	138
NFSERR_ACCESS	13	EACCES	111
NFSERR_EXIST	17	EEXIST	117
NFSERR_NODEV	19	ENODEV	128
NFSERR_NOTDIR	20	ENOTDIR	135
NFSERR_ISDIR	21	EISDIR	123
NFSERR_EINVAL	22	EINVAL	121
NFSERR_FBIG	27	EFBIG	119
NFSERR_NOSPC	28	ENOSPC	133
NFSERR_ROFS	30	EROFS	141
NFSERR_NAMETOOLONG	63	ENAMETOOLONG	126
NFSERR_NOTEMPTY	66	ENOTEMPTY	136
NFSERR_DQUOT	69	EDQUOT	1133
NFSERR_STALE	70	ESTALE	1134

Table 52 contains the return codes that are defined by the NFS version 3 protocol and z/OS UNIX.

Table 52. z/OS NFS return codes between NFS version 3 protocol and z/OS UNIX

NFS version 3 Protocol value	z/OS UNIX value	hex	Description
NFSS3ERR_PERM	EPERM	008B	Operation not permitted
NFS3ERR_NOENT	ENOENT	0081	No such file/directory
NFS3ERR_IO	EIO	007A	I/O error
NFS3ERR_EXIO	ENXIO	008A	No such devices
NFS3ERR_ACCESS	EACCESS	006F	Permission is denied
NFS3ERR_EXIST	EEXIST	0075	The file exists
NFS3ERR_XDEV	EXDEV	0090	Cross device hard link
NFS3ERR_NODEV	ENODEV	0080	No such device
NFS3ERR_NODIR	ENODIR	0087	Not a directory
NFSS3ERR_ISDIR	EISDIR	007B	The file specified is a directory
NFSS3ERR_INVAL	EINVAL	0079	The parameter is incorrect
NFS3ERR_FBIG	EFBIG	0077	The file is to large
NFS3ERR_NOSPC	ENOSPC	0085	No space on device
NFS3ERR_ROFS	EROFS	008D	Read-only file system
NFS3ERR_MLINK	EMLINK	007D	Too many links occurred
NFS3ERR_NAMETOOLONG	ENAMETOOLONG	007E	Filename is too long
NFS3ERR_NOTEMPTY	ENOTEMPTY	0088	Directory not empty

Table 52. z/OS NFS return codes between NFS version 3 protocol and z/OS UNIX (continued)

NFS version 3 Protocol value	z/OS UNIX value	hex	Description
NFS3ERR_DQUOT	EDQUOT	046D	Disk quota exceeded
NFS3ERR_STALE	ESTALE	046E	Stale file handle
NFS3ERR_REMOTE	EREMOTE	046F	Too many levels of remote in path
NFS3ERR_BADHANDLE	EBADF	0071	Bad file handle/descriptor
NFS3ERR_NOT_SYNC	N/A		See note
NFS3ERR_BAD_COOKIE	N/A		See note
NFS3ERR_NOTSUP	ENOSYS	0456	Protocol not supported
NFS3ERR_TOOSMALL	N/A		See note
NFS3ERR_SERVERFAULT	EIO	007A	I/O error
NFS3ERR_BADTYPE	ENXIO	008A	No such device
NFS3ERR_JUKEBOX	EAGAIN	0070	Resource unavailable

Note: If z/OS NFS client provides necessary recovery, then there is no corresponding z/OS UNIX return code.

Appendix C. Sample CHECKLIST data set

The following is a sample CHECKLIST data set.

```
#####  
#  
# OS/390 Network File System Server Sample CHECKLIST #  
# #  
# PROPRIETARY V3 STATEMENT= #  
# LICENSED MATERIALS - PROPERTY OF IBM #  
# THIS MODULE IS "RESTRICTED MATERIALS OF IBM" #  
# 5647-A01 #  
# (C) COPYRIGHT IBM CORPORATION 1998 #  
# SEE IBM COPYRIGHT INSTRUCTIONS #  
# END PROPRIETARY V3 STATEMENT #  
# (C) Copyright IBM Corp. 1998 #  
# #  
#####  
#  
# This data set contains entries for files/directories that are to be  
# exempt from SAF checking even though SAF or SAFEXP is specified as  
# the security option. This file is only used when SAF or SAFEXP is  
# specified for the particular data type (i.e. MVS data, HFS data,  
# data accessed using the public file handle) AND CHECKLIST option is  
# specified as the site attribute. The entries specified here HAVE TO  
# MATCH a subsequent mount point OR be the parent of a subsequent mount  
# point to allow SAF checking to be bypassed for everything underneath  
# that mount point. If the entry does not match a subsequent mount  
# point and it is not the parent of any subsequent mount point then it  
# has no effect whatsoever.  
#  
# For SAFEXP security option, the EXPORTS requirements have to be  
# met first before the CHKLIST requirements are examined. See  
# example below.  
#  
# Proper care must be used to protect this data set.  
#  
# This data set is read during server startup processing. Subsequent  
# changes to this data set will not take effect until the server is  
# restarted. Errors found in the file are sent to the system log  
# and the server terminates.  
#  
# Statement Syntax  
#  
# Entries can be up to 1023 characters long. Lines can be continued  
# by placing a '+' or '\' at the end of the line. A '#' anywhere in the  
# data set indicates a comment that extends to the end of the line.  
# No spaces may appear in the name; however, leading blanks  
# are ignored in new or continuation lines.  
#
```

```

# Entries are specified by a line of the following
# syntax:
#
# Name
#
# where:
#
# Name = Prefix or file name (which should match a subsequent mount
#       point or be the parent of a subsequent mount point to allow
#       SAF checking to be bypassed for everything underneath that
#       mount point)
#
# Note: Anything after the name on the same line will be ignored.
#
# Example:  Given that security of SAFEXP is specified and the exports
#           list and checklist entries are as follows:
#
# Exports:                Checklist:
# WEBNFS.PDS              WEBNFS
# USER1                   USER1.PUBLIC
# /hfs/user1/tmp          /hfs/user1
# /hfs/u/public           /hfs/u/public
# /hfs/bin
#
# The following mount requests will fail because it is not in the
# exports list: (mount points have to be exportable first)
#   WEBNFS
#   /hfs/user1
#
# The following mount requests will succeed and SAF checking will
# be bypassed for everything underneath that mount point (i.e. user
# do not need to do MVSLOGIN for the files/directories underneath
# the mount point):
#   WEBNFS.PDS
#   USER1.PUBLIC.PDS
#   /hfs/user1/tmp/dir1
#   /hfs/u/public

#
# The following mount requests will succeed but SAF checking will
# be enforced:
#   USER1.SOME.PDS
#   /hfs/bin
#
# If 'maplower' was specified in the processing attributes, all
# entries are translated to upper case.
#
# If 'nomaplower' was specified in the processing attributes,
# attention must be given to the case of entries.

WEBNFS                               #
/hfs/tmp

```

Appendix D. NFS system server sample attribute table

You can use the contents of the Figure 12 attribute table file as a NFS server sample.

```
#####  
#  
# OS/390 Network File System Server Sample Attribute Table #  
#  
# PROPRIETARY STATEMENT= #  
# LICENSED MATERIALS - PROPERTY OF IBM #  
# THIS MODULE IS "RESTRICTED MATERIALS OF IBM" #  
# 5647-A01 #  
# (C) COPYRIGHT IBM CORPORATION 1991, 1996 #  
# SEE IBM COPYRIGHT INSTRUCTIONS #  
# END PROPRIETARY STATEMENT #  
# (C) Copyright IBM Corp. 1991, 1998 #  
# (C) Copyright SUN Microsystems, Inc & #  
# Electronic Data Systems Corp. 1988, 1989 #  
# #  
#####  
#  
# This is a prototype site defaults attribute file for the #  
# OS/390 Network File System Server Sample Attribute Table #  
#  
# '#' character starts a comment. Comments can appear anywhere. #  
# White space is ignored when parsing the file. #  
  
# Default values are illustrated in the examples in this file  
  
#  
# Keywords are not case sensitive. 'BLKS' is the same as 'blks' is  
# the same as 'Blks'.  
  
# All time values are in seconds.  
  
#####  
# The following are known as data set creation attributes. #  
#####  
  
# SPACE specifies the amount of primary and secondary space allocated  
# for a new data set. The syntax is:  
#  
# SPACE(PRIMARY,SECONDARY)  
#  
# The secondary field is optional (if omitted, the default is taken).  
#  
# Dimension of allocation is BLKS, TRKS, or CYLS  
# RECS is a synonym for BLKS.  
  
space(100,10), blks
```

Figure 12. NFS system server sample attribute table (Part 1 of 12)

```

# RLSE specifies that unused space should be released from the data
# set the first time that a new data set is closed. For slow clients,
# with long pauses between writes, the RLSE attribute will cause space
# to be released from the primary extent prematurely. Subsequent
# writes will cause secondary space to be allocated.

norlse

# The record format, or RECFM, defines part of the layout of a data
# set: how the records are physically layed out on disk.
#
# Valid RECFM characters are:
#
#     V - Variable Length Records (LRECL defines maximum size of
#         any record)
#     F - Fixed Length Records (LRECL defines the actual length of
#         all records)
#     U - Undefined Length Records
## modified by:
#
#     B - Records are Blocked (BLKSIZE defines the size of the block)
#     S - Spanned for variable length records
#         Standard format for fixed length records
#     M - Machine Control Codes
#     A - ANSI Control Codes
#
# "A" and "S" are mutually exclusive
# "V", "F", and "U" are mutually exclusive
# "S" is not allowed for DSNTYPE(PDS) and DSNTYPE(LIBRARY)
# (refer to DSNTYPE later in this section.)
#
# The BLKSIZE is the size, in bytes, of a physical block on disk.
# BLKSIZE(0) allows the system to choose an optimized block size.
#
# LRECL defines the size, in bytes, of a logical record in the data set.

recfm(vb), blksize(0), lrecl(8196)

# The data set organization can be one of:
#
#     PS           - Physical Sequential
#     DA           - Direct Access
#     INDEXED      - VSAM KSDS data set
#     NONINDEXED   - VSAM ESDS data set
#     NUMBERED     - VSAM RRDS data set
#
# PS is a good organization for NFS usage, and NONINDEXED is the
# corresponding good VSAM data set for NFS (e.g. with AIX client in
# BINARY mode) usage.

dsorg(ps)

# DSNTYPE specifies whether a PDS or a PDSE is to be created when
# the make directory workstation command is issued.
#
# Valid DSNTYPEs are:
#
#     PDS         - Create a Partitioned Data Set.
#     LIBRARY     - Create a Partitioned Data Set Extended.

dsntype(pds)

```

Figure 12. NFS system server sample attribute table (Part 2 of 12)

```

# Number of Directory blocks for PDS allocation

dir(27)

# The MGMTCLAS specifies the management class associated with the
# file creation.
#
# The syntax is:
#
#   mgmtclas(mgmt_class_name)# The VOLUME (or VOL) attribute enables you to specify the volume
# on which to create the specified data set.
#
# The syntax is:
#
#   volume(volser)

# The UNIT attribute enables you to specify the unit on which to
# create the specified data set.
#
# The syntax is:
#
#   unit(unit_name)

# The following attributes are used to control VSAM data set
# creation. They are used only if the DSORG parameter defines
# the data set to be type INDEXED, NONINDEXED or NUMBERED.
#
# Refer to appropriate IBM MVS documentation for a more
# complete description of these and other data set creation
# attributes.

# The KEYS(LENGTH,OFFSET) attribute enables you to define the key
# length and offset for a VSAM INDEXED (KSDS) data set. It is used
# only if DSORG is INDEXED.
#
# Valid range for LENGTH is from 1 to 255.
# Valid range for OFFSET is from 0 to 32760.

keys(64,0)

# The RECORDSIZE(AVERAGE,MAXIMUM) attribute enables a user to define
# the average and maximum record sizes for a VSAM data set.
# These two values must be equal for NUMBERED (RRDS) data sets.
#
# Valid range is from 1 to 32760.

recordsize(512,4K)

# The SPANNED and NONSPANNED attributes define whether VSAM
# records will span control intervals. This option does not affect
# non-VSAM variable length record data sets. Use the 'S' option
# with the RECFM attribute for non-VSAM data sets.

nonspanned

# The SHAREOPTIONS attribute defines the cross region and cross
# system file sharing allowed for a VSAM data set.
#
# Valid range for each argument is from 1 to 4.

shareoptions(1,3)

```

Figure 12. NFS system server sample attribute table (Part 3 of 12)

```

#####
# The following are known as processing attributes.
#####

# There are three timeout types: attributes, reads and writes.
# The various timeout values are used by the system to determine
# when to close and deallocate an inactive data set after the last
# "attribute", "read", or "write" operation.
# The WRITETIMEOUT is usually kept short, because WRITE
# operations result in exclusive locking, and you'll want to release
# the data set. For slow clients, with long pauses between writes,
# you'll want to increase the WRITETIMEOUT value.
#
# Valid range is from "m" to "n"; where "m" is the argument of
# MINTIMEOUT(m) and "n" is the argument of MAXTIMEOUT(n), unless
# NOMAXTIMEOUT is specified. In that case, "n" is 32767.
#
# xxxxTIMEOUT(n) indicates to deallocate the data set n seconds after
# the "xxxx" operation;
# NOxxxxTIMEOUT indicates not to deallocate the data set after
# the "xxxx" operation;
# Where "xxxx" can be "ATTR", "READ", or "WRITE".
#
# e.g. WRITETIMEOUT(1) indicates to deallocate the data set 1 second
# after a write.
# NOWRITETIMEOUT indicates not to deallocate the data set after
# a write.
# READTIMEOUT(90) indicates to deallocate the data set 90 seconds
# after a read if no further activity against it.

attrtimeout(120), readtimeout(90), writetimeout(30)

# Processing may be TEXT or BINARY
#
# BINARY is good for using MVS as a disk farm for PCs and AIX machines
# and offers better performance.
#
# TEXT should be specified if it is necessary to share data sets
# containing textual data with other MVS applications.
#

binary

#
# MAPPED should be specified when a mixed set of data types are
# to be processed on a single mount point. The determination
# of whether the data is to be processed as text or binary
# depends on the rules established in the specified side file.
# If a file extension cannot be mapped to text or binary, then
# the data will be processed according to what has been specified
# as binary or text at the mount level, and finally, the site
# level. If binary or text is specified at the file level,
# the specification overrides the MAPPED specification.
#

```

Figure 12. NFS system server sample attribute table (Part 4 of 12)

```

# The syntax is:
#
# mapped
#

# The end of line terminators are:
#
#      CR, CRLF, LF, LFCR, or NOEOL.
#
# They define the conversion of records to line terminators in TEXT
# mode.
# LF should be used for AIX clients;
# CRLF should be used for PC clients.
# (set through client mounts appropriately).  When TEXT mode is
# specified, LF is the default.

LF

# BLANKSTRIP or NOBLANKSTRIP affects the processing of trailing
# blanks when reading and writing to Fixed Record data sets with
# text processing enabled.

blankstrip

# The MAPLEADDOT attribute turns on mapping of a file name starting
# with a leading "." from a client to a legal leading "$" for a
# MVS data set name.  NOMAPLEADDOT turns off this mapping.

mapleaddot

# The MAPLOWER attribute tells the server to map file names to
# uppercase when received from the client in an NFS request and
# to translate from uppercase to lowercase when returned to the client.
# Keywords are not case sensitive and are unaffected by this option.
#
# The NOMAPLOWER attribute tells the server NOT to do any translation.
# i.e. the server is neither to map uppercase when received from
# the client nor to translate to lowercase when returned to the client.
# All the entries in the EXPORTS file are case sensitive.
# The client MOUNT request must specify the MVS qualifier with
# the correct case to successfully match the EXPORTS file entry.

maplower

# RETRIEVE tells the server to recall a migrated data set on read/write
# access.  NORETRIEVE will force the return of "Device not available"
# error for migrated files.
# RETRIEVE can be coded in the following ways:
# RETRIEVE          - wait for recall if on DASD, nowait otherwise.
# RETRIEVE(WAIT)   - wait for recall.
# RETRIEVE(NOWAIT) - nowait for recall.
# RETRIEVE is the default

retrieve

```

Figure 12. NFS system server sample attribute table (Part 5 of 12)

```

# The FASTFILESIZE attribute tells the server to calculate approximate
# file sizes from available catalog information and disk geometries.
# This approximate size may cause problems with client applications
# since the size is probably inaccurate. The NOFASTFILESIZE may
# result in decreased performance because the server may read a data
# set, applying the defined processing attributes, to determine the
# exact size of the data set as viewed by an NFS client.

```

nofastfilesize

```

# The SETOWNERROOT keyword tells the server to set the user ID in the
# attributes returned to a client for a specified file to 'root' when
# the client is logged on as superuser. SETOWNERNOBODY tells the
# server to set the user ID in the attributes to 'nobody' (-2).

```

setownerroot

```

# You can have the execute bit for plain files on or off by mount
# point. Turn this option on if you plan to store executables
# or shell scripts on the MVS system on a mount by mount basis.
# It should probably always be off in the site file.
#
# EXECUTEBITON will turn on the execute bits (user, group
# and other) for a mount point's files.

```

executebitoff

```

# If the installation intends to customize the translation table,
# a new DD card, NFSXLAT, is required in the NFSS startup proc.
#

```

```

# //NFSXLAT      DD      DSN=dataset_name,DSP=SHR
#

```

```

# Where dataset_name is the name of PDS or PDSE whose
# members are the translation tables.
#

```

```

# The XLAT(member) keyword tells the server which member
# the server is to use as the installation default translation
# table. 'member' is the name of a translation table which
# resides in a PDS or PDSE dataset.
#

```

```

# The syntax is:
#

```

```

# xlat(member)                                     #

```

```

# FILEEXTMAP or NOFILEEXTMAP affects the file extension mapping
# capability. FILEEXTMAP turns on file extension mapping and
# NOFILEEXTMAP turns it off. This option can be specified at
# the file command level. The default is NOFILEEXTMAP.

```

nofileextmap #

```

# SIDEFILE(dsname) specifies the name of the data set that
# contains the rules for file extension mapping purposes.
# If a side file name is specified in the attributes data set
# then it is the default side file for this NFS server.
# A user can also specify another side file name during a MOUNT
# operation to be used along with the default. The mapping rules
# will first be searched in the side file specified during MOUNT
# and then in the default. To allow file extension mapping a
# side file name must be specified either as a default or in the
# MOUNT command. dsname is a fully-qualified MVS data set name
# without quotation marks. SIDEFILE is only specifiable at the
# MOUNT level. See GFSAPMAP for sample mapping side file and
# syntax.

```

Figure 12. NFS system server sample attribute table (Part 6 of 12)

```

# CLN_CC SID(n) specifies the Coded Character Set Identifier(CCSID)
# for the remote mounted file system (NFS client) when text is
# being translated.
# The default (if specified) is 819 (ISO 8859-1 ASCII).
# This attribute applies only to HFS files.
# cln_ccsid(819) #

# SRV_CC SID(n) specifies the Coded Character Set Identifier(CCSID)
# for the local mounted file system (OS/390 NFS Server) when
# a new file is being created.
# This attribute has no effect on the translation of existing
# existing files' data.
# The default (if specified) is 1047 (Latin Open System EBCDIC).
# If this attribute is not specified, new files will continue
# to be created as untagged.
# This attribute applies only to HFS files.
# srv_ccsid(1047) #

#####
# The following are known as site attributes. #
#####

# The following are attributes specifiable ONLY in the site
# file (this file).
#
# Some of these values control internal structures and processing
# within the NFS server. Tuning of these values to improve performance
# should be done incrementally and tested.

# SECURITY attribute control the level of security checking.
# The format of the security keyword is security(mvs,hfs,public)
# where:
# mvs - security option for mvs data access
# hfs - security option for HFS data access
# public - security option for data access with the public
# filehandle
# The first positional parameter is required and the other two
# are optional. When the optional parameters are not specified
# they are assigned the same security as the first parameter.
# Four options can be chosen from. They are:
# NONE - No security checking is performed.
# SAF - SAF checking is performed in line.
# EXPORTS - EXPORTS file is used to check security.
# SAFEXP - Both SAF and EXPORTS file checks are performed.
#
#
# Defaults are SAFEXP for all data accesses.

security(safexp,safexp,safexp) #

# PCNFSD tells the server to start PCNFS server.
# NOPCNFSD tells the server not to start PCNFS server.
# If not specified PCNFSD, default is NOPCNFSD.

nopcnsfd

```

Figure 12. NFS system server sample attribute table (Part 7 of 12)

```

# LEADSWITCH tells the server to return '/' as the first character
#       in each export entry.
# NOLEADSWITCH tells the server not to return '/' as the first character
#       in each export entry.
# If not specified NOLEADSWITCH, default is LEADSWITCH.

leadswitch

# MINTIMEOUT and MAXTIMEOUT set allowable values for
# ATTRTIMEOUT, READTIMEOUT, and WRITETIMEOUT. Specify NOMAXTIMEOUT
# to allow NOATTRTIMEOUT, NOREADTIMEOUT and NOWRITETIMEOUT specification
# by clients.
#
# Valid range is from 1 to 32767.

mintimeout(1)
nomaxtimeout

# The logout time is the time limit on inactivity for a given user
# on a machine. When the limit is reached, the user is automatically
# logged out. The user must then do another "mvslogin" to restart
# the session. The time value is specified in seconds.
#
# You would probably want to set LOGOUT to the TSO timeout value that
# is defined at your site.
#
# MAXTIMEOUT does not affect the LOGOUT site attribute.
#

logout(1800)                # 30 minutes (30 * 60)

# The readdirtimeout is a new timeout attribute to control the
# timeout of the readdir cache used by MVS conventional data
# sets. The timeout value controls how long before the readdir
# results saved in cache are discarded.
#
# Valid range is from 1 to 32,767 seconds.
#
# n can go as low as 1 second but to avoid the possibility of
# client hanging (because of network delays and staled cache),
# n is not recommended to be lower than 5 seconds.
# n may need to be increased if the network is slow and the
# accessed directory has a lot of entries.
# The default readdirtimeout is 30 seconds.

readdirtimeout(30)                #

# The RDRVERF attribute tells the server to do cookie verifier
# checking for NFS version 3 readdir and readdirplus requests.
#
# The NORDRVERF attribute tells the server not to do cookie
# verifier checking for NFS version 3 readdir and readdirplus
# requests.
# The default is NORDRVERF.

nordrverf                #

```

Figure 12. NFS system server sample attribute table (Part 8 of 12)

```

#
# The NFSTASKS(n,m,o) defines the number of NFS tasks (or
# threads) to spawn.
#
# If NFSTASKS(n,m) is specified, then the following is true:
#
# 'n' is the number of subtasks which handle the asynchronous I/O
# operations or short blocking operations (the maximum number of
# concurrent NFSS requests).
# 'm' is the number of subtasks which handle the long blocking
# operations (the maximum number of concurrent NFSS recall and
# HFS requests.). Increase this value if your server supports
# lots of active recall or HFS clients.
#
# Valid range for 'n' is from 1 to 24.
# Valid range for 'm' is from 1 to 24.
# The sum of 'n' plus 'm' must be less than or equal to 25.
#
# If NFSTASKS(n,m,o) is specified, then the following is true:
#
# 'n' is the number of subtasks which handle the asynchronous I/O
# operations or short blocking operations (the maximum number of
# concurrent NFSS requests).
# 'm' is the number of subtasks which handle HFS requests.
# Increase this value if your server supports lots of active
# HFS clients.
# 'o' is the number of subtasks which handle the long blocking
# operations (the maximum number of concurrent NFSS recall
# requests.) Increase this value if your server supports
# lots of active recall.
#
# Valid range for 'n' is from 1 to 24.
# Valid range for 'm' is from 1 to 100.
# Valid range for 'o' is from 1 to 24.
# The sum of 'n' plus 'o' must be less than or equal to 25.

nfstasks(8,16,8) #

# The RESTIMEOUT(n,m) defines resource cleanup timer.
# 'n' is the resource retention period for mounts an
# associated resources which will be removed if the
# have been inactive more than 'n' number of hours.
# 'm' is the time of day to do the cleanup wor
# for mounts and associated resources which
# have been inactive more than 'n' number of hours.
# The time of day is specified as a 24 hour local time value
# The starting time is at least 24 hours from NFSS started up
#
# NFSS will appear slow during this cleanup activity.
# Cleanup will be executed under main task, thus preventing
# any additional work from executing.
# The value of RESTIMEOUT should be set to a value
# such that this cleanup activity will occur while
# NFSS is lightly loaded.
#
# Valid range for 'n' is from 1 to 720.
# If n is set to 0, NFSS will not remove any mount points.
# Note: NFSS keeps all the information for the
# inactive mount points, thus creating long chains
# to be searched.
#

```

Figure 12. NFS system server sample attribute table (Part 9 of 12)

```

# Valid range for 'm' is from 0 to 23.

restimeout(48,0) #

# The CACHEWINDOW attribute limits the number of windows to be used
# for each data set in caching client block writes which are received
# out of order.
# The size of each window is determined by the packet size.
# The suggested value is some small multiple of
# the number of BIODs running on an NFS client.
#
# Valid range is from 1 to 256.

cachewindow(112) #

# The HFS attribute specifies a new HFS file system prefix to be
# imbedded in the mount directory path name. The default value of
# the HFS file system prefix is /hfs. Mount requests received by
# the Network File System beginning with the HFS file system
# prefix value are identified as mount requests for OpenEdition
# MVS. The HFS file system prefix value is not part of the path
# name.
#
# Note: The HFS file system must be mounted locally by OpenEdition
# MVS or the client mount fails.
# hfs(prefix) | nohfs

nohfs #

# The LOGICALCACHE attribute sets the high water mark for all the
# cache windows combined(in bytes).
#
# Valid range is from 1 to 128MB

logicalcache(16M) #

# The BUFHIGH attribute sets the high water mark, in
# bytes, of data buffers before a buffer reclamation take place.
# A higher number means more caching and probably better read
# performance.
#
# Valid range is from 1 to 128MB.
#
# (If BUFHIGH + LOGICALCACHE is larger than the available storage
# in the extended private area (implied by the REGION parameter
# coded in your PROC) at startup, the NFS server will
# shutdown immediately.)

bufhigh(32M) #

# On reaching the BUFHIGH high water mark, a percentage of buffers
# is reclaimed for reuse. This is the PERCENTSTEAL. A higher value
# means a reclaim operation is performed less often, but that
# the cached buffers will be significantly trimmed on each reclaim.
# This can result in a poor read performance, because readahead
# buffers may be stolen.
# Lower values result in more frequent reclaim operations, but the
# cached buffers are not significantly trimmed on each reclaim.
#
# Valid range is from 1 to 99.

```

Figure 12. NFS system server sample attribute table (Part 10 of 12)

```

percentsteal(20)

# The READAHEADMAX value defines the maximum read ahead (in bytes)
# during read processing.
# This reduces the amount of synchronous physical I/O required for
# NFS read requests in sequential processing. It also reduces context
# switching overhead on NFS read requests by allowing more read
# requests to be satisfied directly from the main task.
#
# The number is usually set to 2 to 4 times the common block size
# used for file access (which is recommended at 8K for AIX file
# activity).
#
# Valid range is from 1 to 128K

readaheadmax(16K)

# The MAXRDFORSZLEFT attribute defines the number of physical block
# buffers to be remained after a read-for-size operation.
# These buffers are remained to satisfy potential subsequent
# NFS read requests against the same file.
# The buffers remained are subject to trimming during buffer
# steal operations.
#
# Valid range is from 1 to 1024.

maxrdforszleft(32)

# SMF attribute controls the level of smf support.
# Four options can be chosen from. They are:
# NONE      - No smf records are to be produced.
# USER     - User session smf records are to be produced.
# FILE     - File usage smf records are to be produced.
# USERFILE - Both user session and file usage smf records are
#           to be produced.
# Default is NONE.

smf(none)

# SFMAX(n) where n specifies the maximum size (in kilobytes) of
# allocated storage to hold all of the side files. n is an
# integer from 0 to 2000 (2MB). The default value is 0 and it
# also signifies that no mapping is allowed on this NFS server.
# If SFMAX=0, the specification of sidefile in the attributes
# data set will cause the server to shut down and the
# specification of sidefile in any subsequent MOUNT commands will
# cause the mount to fail as mapping is not allowed on this NFS
# server. If the amount of storage specified can not be obtained
# during server initialization then the server will shut down
# immediately.
#
# The default is SFMAX(0).

sfmax(0)

```

Figure 12. NFS system server sample attribute table (Part 11 of 12)

```

# PUBLIC(legacy_path,hfs_path) specifies the legacy path
# (MVS conventional data) and/or HFS path that is associated with
# the public file handle for WebNFS access. The first path, if
# specified, is the legacy path. The second path is the HFS path
# and must start off with the HFS prefix specified in the HFS()
# keyword. If the first path is not there, a comma must precede
# the second path. If the PUBLIC keyword is specified then one
# of the paths must be specified. The PUBLIC keyword must be
# specified after the HFS() keyword in this site attribute table.
# A LOOKUP request with the public file handle will determine
# which of the two paths it is referring to by the pathname that
# it comes in with. An absolute pathname will tell the server
# which of the path it is referring to by a match on one of the
# paths specified. A LOOKUP request with a relative pathname
# will be taken to be an HFS request if HFS is active (i.e. an
# hfs_path has been provided); otherwise, it is treated as a
# legacy request.
#
# The default is no PUBLIC paths.

# CHECKLIST specifies that the data set pointed to by the CHKLIST
# DD in the startup procedure will be scanned. The data set
# contains a list of entries which should match a subsequent
# mount point or be the parent of a subsequent mount point to
# allow SAF checking to be bypassed for everything underneath
# that mount point. CHECKLIST is only valid if SAF checking is
# the security option for the particular data access; otherwise,
# it is ignored even if specified. See GFSAPCHK in NFSSAMP
# library for sample CHKLIST data set.
# NOCHECKLIST will cause the server not to look at the information
# in the CHKLIST data set even if it is there. This is the
# default.

nochecklist                                     #

# FN_DELIMITER specifies a character to be used instead of a
# comma to delimit the file name from the accompanying attributes.
# This will allow those sites which have UNIX data sets containing
# commas to copy and store their data on the OS/390 NFS Server.
#
# An example of this would be:
#   The attribute data set contains fn_delimiter(;)
#   this changes the default delimiter character from a comma
#   to a semicolon.
#
#   so instead entering ... vi "data_set_name,text,lf"
#
#   you would enter.....vi "comma,in-name;text,lf"
#
# This is a new SITE attribute and it's default is a comma
#
fn_delimiter(,)                                 #

```

Figure 12. NFS system server sample attribute table (Part 12 of 12)

Appendix E. Sample exports data set

Figure 13 contains an example exports data set. The exports data set can be found as GFSAPEXP in the NFSSAMP library.

```
#####  
#  
# z/OS Network File System Server Sample EXPORTS #  
# #  
# PROPRIETARY STATEMENT= #  
# LICENSED MATERIALS - PROPERTY OF IBM #  
# THIS MODULE IS "RESTRICTED MATERIALS OF IBM" #  
# 5695-DF1 #  
# (C) COPYRIGHT IBM CORPORATION 1991, 1996 #  
# SEE IBM COPYRIGHT INSTRUCTIONS #  
# END PROPRIETARY STATEMENT #  
# (C) Copyright IBM Corp. 1991, 1996 #  
# (C) Copyright SUN Microsystems, Inc & #  
# Electronic Data Systems Corp. 1988, 1989 #  
# #  
#####  
#  
# This data set contains entries for directories that may be  
# exported to Network File System Clients. It is used by the  
# server to determine which data sets and prefixes may be  
# accessed by a client, and to write protect data sets on the  
# server provided that the SECURITY site attribute is set  
# to either SECURITY(EXPORTS) or SECURITY(SAFEXP). This file  
# is not used for SECURITY(SAF) or SECURITY(NONE).  
#  
# The mvsnfs.cntl(exports) data set is read during server startup  
# processing. Subsequent changes to this data set will not take  
# effect until the server is restarted or the EXPORTFS command  
# is issued. However, changes to the data set do NOT affect the  
# mounts processed before the server was restarted or before  
# EXPORTFS was issued.  
#  
# Errors found in the file are sent to the system log, and the  
# server continues processing for minor errors such as undefined  
# host names. Termination is brought about if the EXPORTS data  
# set cannot be read or if a syntax error exists. (In the case of  
# the EXPORTFS command, these errors will cause the command to be  
# ignored and processing will continue with the original exports  
# file in effect.)  
#  
# Statement Syntax  
#
```

Figure 13. Sample exports data set

```

# Entries can be up to 4096 characters long. Lines can be continued
# by placing a '+' or '\' at the end of the line. A '#' anywhere
# in the data set indicates a comment that extends to the end of
# the line.
# No spaces may appear in the keywords; however, leading blanks
# are ignored in new or continuation lines.
#
# Entries for a directory are specified by a line of the following
# syntax:
#
#   directory  -ro
#   directory  -rw=clients
#   directory  -access=clients
#
# where:
#
# directory = Prefix or file name
#
# -ro       = Export the directory as read only. If not specified,
#            the directory is exported read/write to every one.
#
# -rw       = Directory exported read/write to specified clients,
#            and read only to everyone else. Separate clients
#            names by colons.
#
# -access   = Give access only to clients listed. Separate client
#            names by colons. This can be further qualified with
#            the "ro" keyword or with an "rw" list.
#
# The keywords "ro" and "rw" are mutually exclusive.
#
# If no options are specified, the default value
# allows any client read/write access to the given directory.
#
#
# If 'maplower' is specified in the default attributes, all
# entries are translated to upper case. High level qualifiers
# specified in the client mount request are translated to
# upper case.
#
# If 'nomaplower' is specified in the default attributes,
# attention must be given to the case of entries. High level
# qualifiers specified in the client mount request are not
# translated to upper case.

mvsnfs                -ro
userid0.mixds         -rw=fsrs001:fslab004:fssun04
userid1               -rw=fsrs001
userid2.nfs.pds       -access=fsrs001:fssun03
userid3.nfs.ps        -access=fslab004
userid4.nfs.another.pds -access=fsrs001,ro
/hfs/u/newproducts

```

Appendix F. Sample startup procedures

The following text contains sample startup procedures that you can use to start the z/OS NFS server and client.

Sample z/OS NFS server startup procedures

Figure 14 contains a sample MVS NFS z/OS NFS server startup procedure. This procedure can be found as GFSAPROC in the NFSSAMP library.

```
//MVS NFS PROC MODULE=GFSAMAIN,PARMS='INFO',
//      SYSNFS=SYS1,STYLE=SYS1,NFSRFX=MVS NFS,TCPIP=TCPIP
//*****
//*
//*z/OS NETWORK FILE SYSTEM SERVER START UP PROC(HDZ11TS)
//*
//* PROPRIETARY STATEMENT=
//* LICENSED MATERIALS - PROPERTY OF IBM
//* THIS MODULE IS "RESTRICTED MATERIALS OF IBM"
//* 5647-A01
//* (C) COPYRIGHT IBM CORPORATION 1989, 1998
//* SEE IBM COPYRIGHT INSTRUCTIONS
//* END PROPRIETARY STATEMENT
//*
//*****
//GFSAMAIN EXEC PGM=&MODULE,
//      PARM='&PARMS',
//      REGION=0M,
//      TIME=1440
//*
//*
//* STEPLIB CONSISTS OF NFS LIB AND LANGUAGE ENVIRONMENT LIBRARIES
//* CHANGE THE NAMES AS APPROPRIATE FOR YOUR INSTALLATION
//* NOTE: EACH OF THESE LIBRARIES MUST BE AUTHORIZED AND
//* LANGUAGE ENVIRONMENT V1R5M0 OR HIGHER IS REQUIRED
//*
//* &TCPIP..TCPIP.DATA IS TCP/IP DATA FILE
//* &SYSNFS..NFS LIB IS z/OS NETWORK FILE SYSTEM SERVER
//* TARGET LIBRARY
```

Figure 14. Sample z/OS NFS server startup procedures

```

//* &SYSLE..SCEERUN IS LANGUAGE ENVIRONMENT RUNTIME LIBRARY
//*
//SYSTCPD DD DISP=SHR,DSN=&TCPIP..TCPIP.DATA
//STEPLIB DD DISP=SHR,DSN=&SYSNFS..NFSLIB
// DD DISP=SHR,DSN=&SYSLE..SCEERUN
//*
//SYSPRINT DD SYSOUT=*
//*****
//*
//* THE SYSDUMP DATASET NEEDS TO BE PRE-ALLOCATED TO THE *
//* DCB SPECIFICATIONS REQUIRED BY SYSDUMP *
//* *
//*****
//SYSDUMP DD DISP=SHR,DSN=&NFSPRF..SYSDUMP
//OUTPUT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//*
//*****
//*
//* NFSLOG1 AND NFSLOG2 DD'S ARE ADDED. *
//* THE TWO LOG DATA SETS NEED TO BE PRE-ALLOCATED *
//* AS SEQUENTIAL FILES. *
//* THE DCB WILL BE SET TO (VB,LRECL=137,BLKSIZE=6144) *
//* ONCE MVS NFS IS STARTED, REGARDLESS OF THE ALLOCATION DCB. *
//* INITIALLY, WE RECOMMEND SPACE=(CYL,(2,5),RLSE) *
//* *
//*****
//NFSLOG1 DD DISP=SHR,DSN=&NFSPRF..LOG1
//NFSLOG2 DD DISP=SHR,DSN=&NFSPRF..LOG2
//*
//* &NFSPRF..CNTL IS A FB LRECL 80 DATASET (PDS OR PS)
//* SAMPLE NFSATTR CAN BE FOUND IN GFSAPATT IN &NFSPRF.NFSSAMP
//* SAMPLE EXPORTS CAN BE FOUND IN GFSAPEXP IN &NFSPRF.NFSSAMP
//*
//NFSATTR DD DISP=SHR,DSN=&NFSPRF..CNTL(NFSATTR)
//EXPORTS DD DISP=SHR,DSN=&NFSPRF..CNTL(EXPORTS)
//*****
//*
//* FHDBASE AND FHDBASE2 ARE *
//* THE MOUNT FILE HANDLE DATABASES. *
//* THEY NEED TO BE PREALLOCATED *
//* AS EMPTY VSAM KSDS DATA SETS. *
//* THEY WILL BE USED ALTERNATELY. *
//* SAMPLE JCL CAN BE FOUND IN &NFSPRF..NFSSAMP(GFSAMHDJ). *
//*****
//FHDBASE DD DISP=SHR,DSN=&NFSPRF..FHDBASE
//FHDBASE2 DD DISP=SHR,DSN=&NFSPRF..FHDBASE2
//*****
//*
//* IF THE INSTALLATION INTENDS TO CUSTOMIZE THE TRANSLATION *
//* TABLE, A NEW DD CARD, NFSXLAT, IS REQUIRED IN THE PROC. *
//* THE TRANSLATION TABLE DATASET NEEDS TO BE PRE-ALLOCATED *
//* AS PDS OR PDSE DATASET. THE FORMAT OF THE DATASET HAS TO *
//* CONFORM TO THE TRANSLATION TABLE FORMAT SUPPORTED BY

```

```

/** TCP/IP FOR z/OS.
/**
/*******
/**NFSXLAT DD DISP=SHR,DSN=&NFSPRFX..XLAT
/**
/*******
/**
/**IF THE INSTALLATION INTENDS TO PROVIDE A CHECKLIST
/** DATASET,A NEW DD CARD, CHKLIST, IS REQUIRED IN THE PROC.
/** THE CHECKLIST DATASET NEEDS TO BE PRE-ALLOCATED.
/** THE FORMAT OF THE DATASET HAS TO CONFORM TO THE RULES
/** OUTLINED IN THE SAMPLE CHECKLIST FILE PROVIDED IN
/** GFSAPCHK IN &NFSPRFX..CHKLIST
/**
/*******
/**CHKLIST DD DISP=SHR,DSN=&NFSPRFX..CHKLIST

```

Sample z/OS NFS client startup procedures

Figure 15 contains a sample MVSNFSC z/OS NFS client startup procedure. This procedure can be found as GFSCPROC in the NFSSAMP library.

```

//MVSNFSC PROC SYSNFS=SYS1,SYSLE=SYS1,NFSPRFX=MVSCCLNT,TCPIP=TCPIP
//*****
/**
/** z/OS NETWORK FILE SYSTEM CLIENT START UP PROC(HDZ11TC)
/**
/*******
//MVSCCLNT EXEC PGM=BPXVCLNY,
//          REGION=0M,
//          TIME=1440
/**
/** STEPLIB CONSISTS OF NFSLIB AND LANGUAGE ENVIRONMENT LIBRARIES
/**          CHANGE THE NAMES AS APPROPRIATE FOR YOUR INSTALLATION
/**          NOTE: EACH OF THESE LIBRARIES MUST BE AUTHORIZED
/**
/** &TCPIP..TCPIP.DATA IS TCP/IP DATA FILE
/** &SYSNFS..NFSLIB IS MVS NETWORK FILE SYSTEM CLIENT TARGET LIBRARY
/** &SYSLE..SCEERUN IS LANGUAGE ENVIRONMENT RUNTIME LIBRARY
/**
//SYSTCPD DD DISP=SHR,DSN=&TCPIP..TCPIP.DATA
//STEPLIB DD DISP=SHR,DSN=&SYSNFS..NFSLIB
//          DD DISP=SHR,DSN=&SYSLE..SCEERUN
/**
//SYSPRINT DD SYSOUT=*

```

Figure 15. Sample z/OS NFS client startup procedures

```

//*****
//*
//* THE SYSDUMP DATASET NEEDS TO BE PRE-ALLOCATED TO THE
//* DCB SPECIFICATIONS REQUIRED BY SYSDUMP
//*
//*****
//SYSDUMP DD DISP=SHR,DSN=&NFSPRFX..SYSDUMP
//OUTPUT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//
//*****
//*
//* NFSCMSG1 AND NFSCMSG2 DD'S ARE
//* THE TWO CLIENT LOG DATA SETS NEED TO BE PRE-ALLOCATED
//* AS SEQUENTIAL FILES.
//* THE DCB WILL BE SET TO (VB,LRECL=137,BLKSIZE=6144)
//* ONCE MVSCLNT IS STARTED, REGARDLESS OF THE ALLOCATION DCB.
//* INITIALLY, WE RECOMMEND SPACE=(CYL,(2,5),RLSE)
//
//*****
//NFSCMSG1 DD DISP=SHR,DSN=&NFSPRFX..LOG1
//NFSCMSG2 DD DISP=SHR,DSN=&NFSPRFX..LOG2

```

Sample startup procedure for the z/OS NFS NSM

Figure 16 contains a sample MVSSTATD z/OS NFS NSM startup procedure. This procedure can be found as GFSAPRST in the NFSSAMP library.

```

//MVSSTATD PROC MODULE=GFSAXSTD
//
//*****
//* MVSSTATD STARTUP PROCEDURE TO START SERVER STATD
//*****
//
//STATD EXEC PGM=&MODULE,
// REGION=0M,
// TIME=1440
//
//*****
//
//* STEPLIB CONSISTS OF STATD/LOCKD LOAD LIBRARY
//* LANGUAGE ENVIRONMENT LIBRARY
//* CHANGE THE NAMES AS APPROPRIATE FOR YOUR INSTALLATION
//*
//* h1q.NFSLIB IS z/OS NETWORK FILE SYSTEM TARGET LIBRARY WHICH*
//* CONTAINS STATD/LOCKD LOAD LIBRARY

```

Figure 16. Sample startup procedure for the z/OS NFS NSM

```

//* h1q.SCEERUN IS LANGUAGE ENVIRONMENT RUNTIME LIBRARY
//* The above libraries(NFSLIB, SCEERUN) must be
//* APF authorized.
//*****
//
//STEPLIB DD DISP=SHR,DSN=h1q.NFSLIB
// DD DISP=SHR,DSN=h1q.SCEERUN
//
//*****
//
//SYSPRINT DD DUMMY

```

Sample startup procedure for the z/OS NFS NLM

Figure 17 contains a sample MVSLOCKD z/OS NFS NLM startup procedure. This procedure can be found as GFSAPRLK in the NFSSAMP library.

```
//MVSLOCKD  PROC MODULE=GFSAXLCK
//*
//*****
//* MVSLOCKD STARTUP PROCEDURE TO START SERVER LOCKD
//*****
//*
//LOCKD     EXEC PGM=&MODULE,
//          REGION=0M,
//          TIME=1440
//*
//*****
//*
//* STEPLIB CONSISTS OF STATD/LOCKD LOAD LIBRARY
//*                   LANGUAGE ENVIRONMENT LIBRARY
//* CHANGE THE NAMES AS APPROPRIATE FOR YOUR INSTALLATION
//*
//* h1q.NFSLIB IS z/OS  NETWORK FILE SYSTEM TARGET LIBRARY WHICH*
//*           CONTAINS STATD/LOCKD LOAD LIBRARY                *
//* h1q.SCEERUN IS LANGUAGE ENVIRONMENT RUNTIME LIBRARY        *
//* The above libraries(NFSLIB, SCEERUN) must be               *
//* APF authorized.                                           *
//*****
//*
//STEPLIB DD  DISP=SHR,DSN=h1q.NFSLIB
//          DD  DISP=SHR,DSN=h1q.SCEERUN
//*
//*****
//*
//SYSPRINT DD  DUMMY
//*
```

Figure 17. Sample startup procedure for the z/OS NFS NLM

Appendix G. Retrieving source code for client enabling commands

Figure 18 shows how you can retrieve the necessary source code to install the client enabling commands on any platform except an AIX, UNIX, or DFSMS/MVS client. See “Installing the client enabling commands” on page 31 for information on retrieving the necessary source code to install the client enabling commands on AIX or UNIX workstation. The z/OS NFS client source code is installed when the z/OS NFS client and TCP/IP are installed. In this example, **mvshost1** is the name of the MVS host and **smith** is an MVS user ID:

```
C> md client          (to create a directory to contain clientcode)
C> cd client
C> ftp mvshost1
Connected to mvshost1
220-FTPSERVE at MVSHOST1, 01:44:24 on 12/02/91
220 Connection closes if idle for more than 5 minutes.
Name (mvshost1:nobody): smith
<Press ENTER key>
331 Send password please.
Password: password

230 smith is logged on.
ftp> get 'mvsnfs.nfstarb(gfsawaxd)' gfsawaxd.c
200 Port request OK.
125 Sending data set MVSNFS.NFSTARB(GFSAWAXD)
250 Transfer completed successfully.
local: gfsawaxd.c remote: 'mvsnfs.nfstarb(gfsawaxd)'
13474 bytes received in 0.30 seconds (50 Kbytes/s)
ftp> get 'mvsnfs.nfstarb(gfsawclt)' gfsawclt.c
200 Port request OK.
125 Sending data set MVSNFS.NFSTARB(GFSAWCLT)
250 Transfer completed successfully.
local: gfsawclt.c remote: 'mvsnfs.nfstarb(gfsawclt)'
22914 bytes received in 5.032100 seconds (51 Kbytes/s)
ftp> get 'mvsnfs.nfstarb(gfsawlin)' gfsawlin.c
200 Port request OK.
125 Sending data set MVSNFS.NFSTARB(GFSAWLIN)
250 Transfer completed successfully.
local: gfsawlin.c remote: 'mvsnfs.nfstarb(gfsawlin)'
18630 bytes received in 0.66 seconds (28 Kbytes/s)
ftp> get 'mvsnfs.nfstarb(gfsawlou)' gfsawlou.c
200 Port request OK.
125 Sending data set MVSNFS.NFSTARB(GFSAWLOU)
250 Transfer completed successfully.
local: gfsawlou.c remote: 'mvsnfs.nfstarb(gfsawlou)'
11759 bytes received in 0.73 seconds (16 Kbytes/s)
ftp> get 'mvsnfs.nfstarb(gfsawmcl)' gfsawmcl.c
```

Figure 18. Retrieving source code for client enabling commands (Part 1 of 2)

```

200 Port request OK.
125 Sending data set MVS NFS.NFSTARB(GFSAWMCL)
250 Transfer completed successfully.
local: gfsawmcl.c remote: 'mvs nfs.nfstarb(gfsawmcl)'
10848 bytes received in 0.66 seconds (16 Kbytes/s)
ftp> get 'mvs nfs.nfstarb(gfsawmou)' gfsawmou.c
200 Port request OK.
125 Sending data set MVS NFS.NFSTARB(GFSAWMOU)
250 Transfer completed successfully.
local: gfsawmou.c remote: 'mvs nfs.nfstarb(gfsawmou)'
28508 bytes received in 0.18 seconds (16 Kbytes/s)
ftp> get 'mvs nfs.nfstarb(gfsawsha)' gfsawsha.c
200 Port request OK.
125 Sending data set MVS NFS.NFSTARB(GFSAWSHA)
250 Transfer completed successfully.
local: gfsawsha.c remote: 'mvs nfs.nfstarb(gfsawsha)'
69433 bytes received in 1.65 seconds (42 Kbytes/s)
ftp> get 'mvs nfs.nfstarb(gfsawmnt)' gfsawmnt.h
200 Port request OK.
125 Sending data set MVS NFS.NFSTARB(GFSAWMNT)
250 Transfer completed successfully.
local: gfsawmnt.h remote: 'mvs nfs.nfstarb(gfsawmnt)'
7080 bytes received in 0.12 seconds (58 Kbytes/s)
ftp> get 'mvs nfs.nfstarb(gfsawsho)' gfsawsho.h
200 Port request OK.
125 Sending data set MVS NFS.NFSTARB(GFSAWSHO)
250 Transfer completed successfully.
local: gfsawsho.h remote: 'mvs nfs.nfstarb(gfsawsho)'
5286 bytes received in 0.13 seconds (41 Kbytes/s)
(Note: Make sure that the tabs key in the gfsawmak makefile will not
      be translated into a space key for the following step.)
ftp> get 'mvs nfs.nfstarb(gfsawmak)' makefile
200 Port request OK.
125 Sending data set MVS NFS.NFSTARB(GFSAWMAK)
250 Transfer completed successfully.
local: makefile remote: 'mvs nfs.nfstarb(gfsawmak)'
8234 bytes received in 0.59 seconds (14 Kbytes/s)
ftp> quit
221 Quit command received. Goodbye.
C>
C> (Port commands source code to be compiled in your client
workstation.)
C> (Set up the path to the "client" directory for the users to use
the executable code of mvlogin, mvlogout or mvlogut,
and showattr commands.)
C> mvlogin mvshost1 smith (MVS NFS must be operational
Password required on host side.)
GFS A973A Enter MVS password:
GFS A955I smith logged in ok.
C>

```

Figure 18. Retrieving source code for client enabling commands (Part 2 of 2)

Appendix H. Using the PCNFSD protocol

The PCNFSD protocol enables clients to access MVS files without issuing the **mvslogin** and **mvslogut** commands. Check with your workstation administrator to see if your clients have PCNFSD support.

Accessing data with PCNFSD

The Network File System creates a unique UID and GID for each client, and the UIDs and GIDs are consistent within a session. For example, the server might return the UID and GID set (100,100) for one user, and (101,100) for the next user.

If the client maintains the UID and GID for the mount point base, you should use the PCNFSD requests rather than the **mvslogin** and **mvslogut** client enabling commands. However, if the client does not maintain the UID and GID for the mount point base, you should use the **mvslogin** or **mvslogut** client enabling commands rather than PCNFSD requests.

If you issue the **mount** command, which issues a PCNFSD request (as it is implemented on some platforms), to do the authentication, you should not use the **mvslogin** or **mvslogut** client enabling commands. For example, this applies to IBM TCP/IP Version 2.1 for DOS.

If the client's MVS user ID, which was authenticated by a PCNFSD request via a **mount** command, is logged off by the server due to timeout or server restart, then the client needs to issue the **umount** and **mount** commands to log on again.

See *z/OS Network File System User's Guide* for information about how clients use PCNFSD.

Accessing z/OS UNIX files

For z/OS UNIX file access, the user must first have a TSO user ID defined to RACF with an OMVS segment defining the UID and GID. When the PCNFSD authenticate request is received, the Network File System uses the UID and GID associated with the user's RACF profile OMVS segment, to return in the client's UID and GID credentials. It is recommended that the RACF installation preserve a unique network UID and GID that each user is associated with throughout the network. If there is no OMVS segment for a user, the client's UID and GID credentials are assigned as presented in the following paragraphs.

If **security(saf)** or **security(safexp)** is specified in the site attributes and the client user does not have a valid OMVS segment, a system assigned UID and GID is returned in the client user credentials.

If **security(none)** or **security(exports)** is specified in the site attributes, a system assigned UID and GID is returned in the client user credentials.

Starting the PCNFSD server

Specify **pcnfsd** or **nopcnfsd** in the attributes data set to control whether to start the PCNFSD server.

PCNFSD protocols supported

You can use either version 1 or 2 of the PCNFSD protocol with the Network File System.

Using version 1 of the PCNFSD protocol

The PCNFSD Version 1 protocol is specified in *Developer's Specification Protocols for X/Open PC Interworking (PC)NFS*. The Network File System supports procedures 0 and 1 of the version 1 PCNFSD protocol.

Procedure 0: do nothing

Procedure 0 (NULL) does nothing.

Procedure 1: perform user authentication

The input parameters for this procedure are the MVS user ID and password. The output parameters are the return code, UID and GID. If the MVS user ID and password are verified successfully (return code=0), the corresponding UID and GID values are returned.

The caller receives one of these return codes:

- **Return code = 0** - Authentication successful
 - **Return code = 2** - Authentication failed
-

Using version 2 of the PCNFSD protocol

NFS supports procedures 0, 1, and 13 of the PCNFSD Version 2 protocol.

Procedure 0: do nothing

Procedure 0 (NULL) does nothing.

Procedure 1: give descriptions

Procedure 1 describes how the NFS supports each procedure of the PCNFSD version 2 Protocol.

The input parameters are the version information and comments from the client platform. The NFS ignores the information in the input parameters.

The output parameters are the version information, comments, number of procedures in the PCNFSD version 2 protocol, and a list of descriptions about how the NFS supports each procedure in this PCNFSD version 2 protocol.

There are three values to represent how the server supports the procedures:

- **-1** - The procedure is not supported
 - **100** - The procedure gets quick service from the server
 - **2000** - The procedure takes more time to complete the service from the server
-

Procedure 13: perform user authentication

The input parameters are *local host name*, *encoded user name*, *encoded password*, and *comments from the requestor*. The NFS ignores the local host name and comments from the requestor, and decodes the user name and password in the input parameters. The NFS does authentication based on the user name and password.

The output parameters are:

There are only two possible return codes that can be sent back to the caller:

- **Return code = 0** - Authentication successful
 - **Return code = 2** - Authentication failed
-

UID and GID

If user name and password are verified successfully (return code = 0), the corresponding UID and GID values are returned.

Number of alternate GIDs

Because the NFS does not support alternate GIDs, this field is set to zero.

Pointer to a list of the alternate GIDs

Because the NFS does not support alternate GIDs, this field is set to zero.

Logon Home Directory

Because the NFS does not support logon home directory, this field is zero.

Default umask

Because the NFS does not support default umask, this field is set to zero.

Comments sent to the requestor

If the return code is 2, this comment field contains the reason why the request failed.

Appendix I. Handling of the file size value

Many NFS procedures (such as `nfs_lookup` and `nfs_getattr`) in the NFS protocol require the file size to be returned. This appendix explains some performance and accuracy considerations in obtaining the file size value.

The meaning of the file size value that is returned by the NFS and how fast the file size is returned depends on the following conditions:

- Whether you use text or binary processing mode
- The type of MVS data set being accessed
- If the data set is system-managed
- If you use **fastfilesize** processing

Storage of the file size value

How the file size value is stored affects how quickly files are accessed and depends on the type of MVS data set used.

System-managed PS, VSAM, and PDSE data sets

For system-managed PS, VSAM KSDS, VSAM ESDS, VSAM RRDS, and PDSE data sets, the file size value is stored on DASD and is returned quickly.

Text and binary file size are saved on non-volatile storage (DASD) and maintained by the server for these data set types:

- Physical sequential (including striped)
- VSAM ESDS
- VSAM KSDS
- VSAM RRDS
- PDSE members

These data sets must be SMS managed. When the NFS accesses a data set for the first time, it performs a read-for-size to get the text or binary file size and stores this value on DASD. Subsequent file size requests from clients do not cause the server to read for size, thus improving performance. However, when the data set is modified outside the server by a non-NFS application (for example, by the TSO editor), the stored file size could be incorrect. When the data set is accessed again by the server, read-for size must be done to determine the correct file size.

Migrated system-managed data sets

DFSMS/MVS allows data set attribute accessibility for SMS managed data sets, without having to recall the data set if the data set is migrated under DFSMS/MVS V1R3. Supported data set types are SMS managed PS, VSAM ESDS, VSAM KSDS, VSAM RRDS, PDS, and PDSE. Migrated PDS/PDSE members are not supported.

The z/OS NFS server is able to obtain the attributes of a supported SMS managed migrated data set without recalling the data set. Attributes such as the record format and file size are saved to DASD. Subsequent file size requests do not cause a recall of the supported SMS managed migrated data set, thus improving performance. However, when the data set is modified outside the server by a non-Network File System application (for example, by the TSO editor) before it was migrated, the stored file size could be incorrect. When the data set is accessed again by the server, a recall must be done to determine the correct file size.

Non-system-managed, PDS, and direct access data sets

The file size value for non-system-managed data sets, PDS members, and direct access (DA) data sets is cached in virtual storage until timeout but not written to DASD. Therefore, for these types of MVS data sets, the file size value is regenerated after the file is closed or after the server is restarted.

How the file size value is generated

When a file is first accessed (for example with **ls -l** or **dir**), usually the entire file is read to determine its size, except for **recfm(f)** or **recfm(fbs)** where the binary size can be computed without reading the file. If the file is a system-managed PS, VSAM, or PDSE member, both binary and text file sizes are stored on DASD, so that subsequent file size requests do not require the file to be read.

Binary file size can be quickly generated by using **recfm(f)** or **recfm(fbs)** to specify a fixed-length record format for the MVS data set. With this format type, the server pads the last logical record with binary zeros in binary mode processing, because MVS always expects complete logical records. If the application tolerates these zeros, using **recfm(f)** or **recfm(fbs)** allows the binary size to be computed quickly because the number of bytes can be computed from the number of blocks, which is stored by MVS.

If you need the exact file size and are using binary mode processing, map it to a variable-format, sequential data set on DASD so that the NFS does not need to pad a partially filled last MVS logical record to a record boundary.

For reading small files or the beginning of files, the read-for-size might not add any processing time. As the file is being read for size, the beginning of the file is stored in the buffers set aside by the **maxrdforszleft** site attribute, until the buffers are full. When the application reads the beginning of the file, this read is fast because it reads directly from the buffer.

MVS stores the number of blocks (rather than the number of bytes) in an MVS file. For most files, therefore, without reading the entire file, the NFS can only give an estimate of the number of bytes in the file, not the exact number of bytes in the file. Even when the server could get the exact byte count without reading the file, the file size could change depending on the file's processing attributes.

For example, selecting **text** mode processing introduces line terminators such as `'lf'`, `'crlf'`, or `'\n'` into the file, thus changing the perceived size of the file. As another example, suppose you select **text** mode processing with blank stripping enabled on a fixed-length record format file. That causes the server to remove trailing blanks from each record, again changing the perceived size of the file. In these examples, when you first request a file, the server must read the entire file to determine its exact size in bytes.

Using fastfilesize to avoid read-for-size

If you can use an approximate file size for a PDS, PDSE, DA, or non-system-managed data set, you can specify the **fastfilesize** attribute to improve performance. With this attribute, the server estimates the size without opening and reading the entire file.

PDS members - For PDS and PDSE members, the **fastfilesize** attribute gets the file size from SPF statistics if they exist; otherwise, a zero file size is returned.

DA data sets - For DA data sets, an approximate file size is calculated based on the device characteristics, the number of disk tracks in use, and the block size of the data set.

VSAM - For non-system-managed VSAM data sets, the estimated size using **fastfilesize** is zero. Therefore, **cat** or **vi** won't show any data.

The **fastfilesize** attribute speeds up data set access by calculating approximate file sizes during data set access. Use this only when you are browsing through files (using the **ls** UNIX command for example) because some commands (such as **cp** or **copy**) might not work correctly if **fastfilesize** is set. When reading or modifying a data set, the **nofastfilesize** attribute should be used to ensure accurate results.

Nofastfilesize

When you use the default, **nofastfilesize** attribute, the NFS reads the entire file or member to get the file size. It stores the file size value in cache until timeout. If the server's default has been changed to **fastfilesize**, you can still use the **nofastfilesize** attribute to override it. For example:

```
$ ls -l "filename,nofastfilesize"
```

Using this attribute might cause a delay when first accessing very large data sets.

Note: When directly mounting on a fully qualified data set name and **nofastfilesize** is specified, the server must return the mount size as part of getting the attributes for the mount. This can slow down the completion of the mount command.

Appendix J. Handling of the time stamps

UNIX file attributes define the following time stamps:

atime - The last time the file was accessed (read)

mtime - The last time the file was modified (write)

ctime - The last time the file status was changed (chmod)

The NFS handles time stamps differently for these types of data sets:

- System-managed PS data sets and system-managed VSAM data sets
- Direct Access data sets and non-system-managed PS data sets
- Non-system-managed VSAM data sets
- PDS and PDSE members

Time stamps for system-managed VSAM and PS data sets

For system-managed PS data sets and system-managed VSAM data sets, *atime* and *mtime* are fully maintained, and the *ctime* is set to the *mtime*.

Time stamps for non-system-managed PS and DA data sets

For non-system-managed physical sequential (PS) and direct access (DA) data sets, consider the following conditions:

- How time stamps are stored.
- The requirements of your workstation programs.
- The type of multiple virtual system (MVS) data set used to store the file.

Storing time stamps

For non-system-managed physical sequential (PS) and direct access (DA) data sets, the Network File System temporarily stores the time stamps in virtual storage, but not on direct access storage device (DASD). These cached attributes are purged when the file times out and closes or when the server is restarted. When the file is accessed again, the time stamps are re-generated.

Client program requirements

Some workstation-based utilities (such as **make**) rely on date and time stamps to determine whether to recompile. For example, **make** checks the update time of the object file with the source file and recompiles if the source has been updated. Before storing these types of files using the MVS server, examine them before moving them to ensure that these attributes are unimportant. In an environment which relies on such utilities, use system-managed PS data sets.

Generating time stamps

This is how the NFS generates *atime* and *mtime* for non-system-managed PS and DA data sets from the MVS dates:

```
atime
= mtime = reference_date + time_increment
ctime = creation_date + time_increment
```

time_increment is either the server local time or 23:59 hours. If *reference_date* or *creation_date* is equal to the server local date, the server local time is added. Otherwise, a fixed value of 23 hours and 59 minutes is added.

If *reference_date* = 0 (that is, the file has not yet been referenced), *atime* and *mtime* are set equal to *ctime*.

Time stamps for non-system-managed VSAM data sets

The time stamps for these types of data sets are set to the current time.

Time stamps for PDSs and PDSEs

An MVS PDS data set can act as a UNIX directory. Members of the PDS are files within the UNIX directory. When the directory is accessed by the client, the UNIX times are expected for each file.

Ordinarily, MVS does not maintain time stamps for members of a PDS. The UNIX time stamps here are generated from the MVS creation and reference dates of the PDS data set containing the members. This is how the time stamps for PDS members are generated:

```
atime
= mtime = reference_date + time_increment
ctime = creation_date + time_increment
```

ISPF is an MVS product that does maintain some additional statistics for each member. They include the creation date and the last modification date and time.

If the ISPF time stamps are present for a PDS member, this is how the server generates the time stamps and initializes the UNIX times:

```
atime = mtime = modification_date + modification_time
ctime = ISPF_creation_date + time_increment
```

time_increment is either server local time or 23:59 hours as described for non-VSAM data sets.

The server also creates new ISPF statistics for PDS members created by the clients. The ISPF statistics are created even if existing members do not have statistics.

The time stamp information is saved in the PDS directory according to ISPF conventions. If *STATS=ON* was specified when the member was created, the server uses them to get more accurate attributes. Even if *STATS=ON* was not specified originally, the server writes back new time stamp information if the member is modified from the workstation.

Time stamp generation for a PDSE member is identical to that of a PDS with one exception. Accurate ***mtime*** of a PDSE member is returned to a client as the result of a file attribute request for that PDSE member.

Setting time stamps

NFS clients can issue SETATTR requests to set the *atime* and *mtime* for a system-managed PS or VSAM data set. For PDSE members, setting *mtime* is allowed, but setting *atime* is not supported. PDSE member *mtime* is also maintained by PDSE access methods, so it is modified when a TSO user modifies the PDSE member.

Appendix K. Sample procedure to generate an SMF report (GFSAPSMF)

Figure 19 contains a sample SMF procedure that can be found in the NFSSAMP library.

```
GFSAPSMF TITLE 'GFSAPSMF - SMF REPORT SAMPLE ROUTINE'
/******
**/
**/ $MOD(GFSAPSMF)  COMP(5695DF121)
**/
**/ MODULE NAME: GFSAPSMF
**/
**/ DESCRIPTION: Generate SMF Report
**/
**/ STATUS: VERSION 1 RELEASE 2
**/
**/ COPYRIGHT:
**/Proprietary V2 Statement
**/Licensed Materials - Property Of IBM
**/"Restricted Materials of IBM"
**/5695-DF1 (C) COPYRIGHT 1994  IBM CORP.
**/End Proprietary V2 Statement
**/
**/ FUNCTION:  Generate SMF Report for Record Type 42 Subtype 7, 8:
**/             Subtype 7: Files statistics
**/             Subtype 8: Session statistics
**/
**/ PROCESSING:
**/
**/ LOGIC:
**/     1. Get work area storage.
**/     2. Initializations: open I/O files, pointers, counters, etc.
**/     3. Read input file until EOF.
**/        a. Process record type 42 subtype 7 and 8
**/     4. When EOF, print total counts.
**/     5. Free work area storage.
**/     6. Return to caller.
**/
**/ ERROR PROCESSING:  None.
**/
```

Figure 19. Sample procedure to generate an SMF report

```

/** NOTES: *
/** *
/** NON-REENTRANT *
/** XAX CONSIDERATIONS: AMODE(31) RMODE(24) *
/** DEPENDENCIES: None. *
/** RESTRICTIONS: None. *
/** REGISTER CONVENTIONS: *
/** Register 0 - 6 : General purpose *
/** Register 7 : Subroutine return address *
/** Register 8 : Work area pointer *
/** Register 9 : SMF 42 Record pointer *
/** Register 10 : Subroutine's parameter *
/** Register 11 : Second base register *
/** Register 12 : First base register *
/** Register 13 : Save area register *
/** Register 14 - 15: General purpose *
/** Floating point FPR0 : Used for conversion from floating point *
/** to fixed. *
/** SERIALIZATION: None. *
/** PATCH SPACE: None. *
/** *
/** MODULE TYPE: CSECT *
/** *
/** PROCESSOR: HIGH LEVEL ASSEMBLER *
/** ATTRIBUTES: *
/** TYPE: CSECT *
/** PRIMARY ASID: Same as HOME *
/** SECONDARY ASID: Same as HOME *
/** HOME ASID: Current *
/** MODE: Task *
/** KEY: 8 *
/** STATE: Problem *
/** LOCATION: Private *
/** Reentrant: No *
/** *
/** ENTRY POINTS: *
/** 1. GFSAPSMF *
/** PURPOSE: Generate SMF Report *
/** LINKAGE: GFSAPSMF *
/** *
/** INPUT: INPTDD DCB - INDD DD card pointing to an input PS file *
/** with recfm(vbs). See NOTE (4). *
/** OUTPDD DCB - OUTDD DD card pointing to an output file. *
/** See NOTE (4). *
/** *
/** OUTPUT: OUTPDD - Output dataset containing the SMF report. *
/** *
/** EXIT_NORMAL: Return. *
/** *
/** EXIT_ERROR: Return. *

```

```

**/
**/ EXTERNAL REFERENCES:
**/ ROUTINES: None.
**/ DATA AREAS: IHADCB
**/ CONTROL BLOCKS: None.
**/ SERVICES:
**/ CLOSE - Close dataset
**/ OPEN - Open dataset
**/ GET - Read input
**/ PUT - Write output
**/ STORAGE OBTAIN/FREE - Get/free storage
**/
**/ MACROS:
**/ 1. GFSAUSMF - SMF Record type 42 subtype 7,8.
**/ 1. DCBD - DCB Mapping.
**/
**/ TABLES: None.
**/
**/ NOTES: 1) In order for SMF to start recording type 42 records
**/ the SYS(TYPE(...)) statement in the SMFPRMxx parmlib
**/ member must be modified to include type 42 record.
**/ For example, before:
**/ SYS(TYPE(0:32,70:80,90,128:134,148:151))
**/ and after type 42 is added:
**/ SYS(TYPE(0:32,42,70:80,90,128:134,148:151))
**/ 2) Once the records are collected by SMF and stored in
**/ SYS1.MANx, the data must be dumped to a sequential
**/ file with recfm(vbs). Refer to the MVS/ESA System
**/ Management Facilities (SMF) manual for details.
**/ Here is sample JCL to perform this:
**/
**/ //SMFDUMP JOB MSGCLASS=A,MSGLEVEL=(1,1),TIME=1440
**/ //*****
**/ //** Dump SMF data to a sequential dataset.
**/ //** If no DCB attributes specified, SMF will apply the
**/ //** following defaults to the output dataset:
**/ //** LRECL=32760, BLKSIZE=4096, RECFM=VBS.
**/ //*****
**/ //STEP1 EXEC PGM=IFASMFDP,REGION=2M
**/ //SYSPRINT DD SYSOUT=*
**/ //DUMPIN DD DSN=SYS1.MANX,DISP=SHR
**/ //DUMPOUT DD DSN=USER1.SMFDATA,
**/ // DISP=(NEW,CATLG,DELETE),
**/ // UNIT=SYSDA,SPACE=(CYL,(5,10))
**/ //SYSIN DD *
**/ INDD(DUMPIN,OPTIONS(DUMP))
**/ OUTDD(DUMPOUT,TYPE(42(7:8)))
**/

```

```

**      3) Assemble and linkedit GFSAPSMF. GFSAPSMF mapping      *
**      macro is required during the assembly step.              *
**      4) GFSAPSMF is invoked via the following sample JCL:     *
**      See note (2) for how to obtain SMF data for INDD.       *
**      The output dataset needs an LRECL of 200 or more.       *
**                                                                *
**      //GFSAPSMF JOB MSGCLASS=A,MSGLEVEL=(1,1),TIME=1440      *
**      //*****                                                *
**      //* Invoking GFSAPSMF report generator                    *
**      //* USER1.LOAD in STEPLIB is the loadlib dataset         *
**      //* generated from step (3) above.                       *
**      //*****                                                *
**      //EXTRACT EXEC PGM=GFSAPSMF,REGION=6144K                *
**      //STEPLIB DD DSN=USER1.LOAD,DISP=SHR                     *
**      //SYSOUT DD SYSOUT=*                                     *
**      //SYSPRINT DD SYSOUT=*                                  *
**      //INDD DD DSN=USER1.SMFDATA,DISP=SHR                    *
**      //OUTDD DD DSN=USER1.SMF.REPORT,DISP=SHR,               *
**      // DCB=(DSORG=PS,RECFM=FM,LRECL=200)                   *
**                                                                *
**      //*****                                                *

```

```

*
*      TITLE 'GENERAL PURPOSE REGISTERS ASSIGNMENTS'
R0      EQU  0          GENERAL PURPOSE REGISTER 0
R1      EQU  1          GENERAL PURPOSE REGISTER 1
R2      EQU  2          GENERAL PURPOSE REGISTER 2
R3      EQU  3          GENERAL PURPOSE REGISTER 3
R4      EQU  4          GENERAL PURPOSE REGISTER 4
R5      EQU  5          GENERAL PURPOSE REGISTER 5
R6      EQU  6          GENERAL PURPOSE REGISTER 6
R7      EQU  7          SUBROUTINE RETURN ADDRESS
R8      EQU  8          DATA POINTER
R9      EQU  9          SMF 42 RECORD POINTER
R10     EQU 10          LENGTH OF PARM PASSED TO HEX2CHAR
R11     EQU 11          SECOND BASE REGISTER
R12     EQU 12          FIRST BASE REGISTER
R13     EQU 13          SAVE AREA PTR
R14     EQU 14          GENERAL PURPOSE REGISTER 14
R15     EQU 15          GENERAL PURPOSE REGISTER 15
FPR0    EQU  0          FLOATING POINT REGISTER 0
GFSAPSMF CSECT
GFSAPSMF AMODE 31
GFSAPSMF RMODE 24
STM R14,R12,12(R13)  STANDARD LINKAGE
LR R12,R15           ESTABLISH BASE REGISTER
USING GFSAPSMF,R12
USING GFSAPSMF+4095,R11

```

```

        LA    R11,4095(R12)    ESTABLISH 2ND BASE REGISTER
        B     SMFXB            BRANCH AROUND ASSEMBLER CONSTANTS
        DC    CL8'GFSAPSMF'    MODULE NAME
        DC    CL8'&SYSDATE'    DATE ASSEMBLED
        DC    CL8'&SYSTIME'    TIME ASSEMBLED
SMFXB  DS     0H
        ST    R13,SAVEAREA+4   SAVE BACKWARD POINTER
        LA    R0,SAVEAREA      GET SAVEAREA ADDRESS
        ST    R0,8(R13)       SET FORWARD POINTER
        LA    R13,SAVEAREA     SET SAVEAREA ADDRESS
* SET UP COMMON COMMUNICATION AREA
*
        STORAGE OBTAIN,LENGTH=PARMSZ,ADDR=(R8) GET PARMLIST STORAGE
        USING PARMLIST,R8
*
*
*****
* INITIALIZATIONS
* 1. OPEN I/O FILES.
* 2. PRESET COUNTERS.
* 3. BLANK OUT BUFFERS.
* 4. INITIALIZE POINTERS.
*****
        BAL   R7,OPENFILE      OPEN I/O FILES
        SLR   R2,R2            RESET READ COUNTER
        ST    R2,RECCNT
        SLR   R5,R5            RESET TYPE 42 SUBTYPE 7,8 COUNTER
        ST    R5,S78CNT
        MVI   IRECORD,X'40'    CLEAR INPUT BUFFER
        MVC   IRECORD+1(BUFFLEN-1),IRECORD
        MVI   BUFFER,X'40'    CLEAR OUTPUT BUFFER
        MVC   BUFFER+1(BUFFLEN-1),BUFFER
        USING SMF42,R9        POINT TO INPUT RECORD
*****
* READ UNTIL EOF
* PROCESS SUBTYPE 42 SUBTYPE 7 AND 8 ONLY.
*****
LOOP   DS     0H
*
        GET   INPTDD          READ RECORD
        L     R2,RECCNT
        LA    R2,1(R2)        NUMBER OF ALL RECORDS READ IN
        ST    R2,RECCNT
        LR    R9,R1           POINT TO SMF 42 RECORD
        CLI   SMF42RTY,X'2A'  IS IT TYPE 42?
        BNE   LOOP
** RECORD TYPE 42. BUT ONLY PROCESS SUBTYPE 7 OR 8.
        CLC   SMF42STY(2),=H'0007' IS IT SUBTYPE 7 ?
        BE    HEAD42
        CLC   SMF42STY(2),=H'0008' IS IT SUBTYPE 8 ?
        BNE   LOOP

```

```

HEAD42  DS   0H
        L    R5,S78CNT
        LA   R5,1(R5)          NUMBER OF SUBTYPE 7 AND 8 PROCESSED
        ST   R5,S78CNT
        BAL  R7,HHEAD          PROCESS TYPE+SUBTYPE HEADERS FIRST
        CLC  SMF42STY(2),=H'0007'  SUBTYPE 7
        BNE  SUBTYPE8
        BAL  R7,ST7            PROCESS SUBTYPE 7
        B    LOOP
SUBTYPE8 DS   0H
        BAL  R7,ST8            PROCESS SUBTYPE 8
        B    LOOP            CONTINUE

```

*

* NO MORE RECORDS TO PROCESS.
PRINT TOTAL COUNTS.

```

EODEOF  DS   0H
        MVI  BUFFER,X'40'      CLEAR OUTPUT BUFFER
        MVC  BUFFER+1(BUFFLEN-1),BUFFER
        L    R2,RECCNT        GET TOTAL RECORDS READ
        CVD  R2,PKD
        UNPK RDCNT,PKD
        MVZ  RDCNT+4(1),RDCNT+3
        MVC  LABEL,RECREAD    PRINT TOTAL RECORDS READ
        PUT  OUTPDD,BUFFER    WRITE TOTAL COUNT MSG
        L    R5,S78CNT        GET TOTAL ACTUAL PROCESSED COUNT
        CVD  R5,PKD
        UNPK RDCNT,PKD
        MVZ  RDCNT+4(1),RDCNT+3
        MVC  LABEL,T42ST78    PRINT TOTAL TYPE 42 ST 7,8 PROCESSED
        PUT  OUTPDD,BUFFER    WRITE TOTAL COUNT MSG
        PUT  OUTPDD,SMFENDM    WRITE COMPLET MSG

```

*

* CLOSE I/O FILES

```

        CLOSE (OUTPDD)        CLOSE OUTPUT FILE
        CLOSE (INPTDD)       CLOSE INPUT FILE

```

*

* RETURN TO CALLER

```

EXIT    DS   0H
        STORAGE RELEASE,LENGTH=PARMSZ,ADDR=(R8)  FREE COMMON STOR
        L    R13,SAVEAREA+4
        LM   R14,R12,12(R13)
        SLR  R15,R15
        BR   R14

```

```

*
*****
#####* END OF MAIN PROGRAM *#####
*****
*
      EJECT
*****
* SUBROUTINE: OPENFILE
* FUNCTION  : OPEN I/O FILES
* INPUT    : INPTDD DCB - INPUT DCB
*          : OUTPDD DCB - OUPUT DCB
* OUTPUT   : FILES OPENED
*****
OPENFILE DS 0H
      OPEN  INPTDD
      OPEN  (OUTPDD,(OUTPUT))
      PUT   OUTPDD,HEADERM  TITLE
      PUT   OUTPDD,NULLREC
      BR    R7          RETURN
*****
* SUBROUTINE: HHEAD
* FUNCTION  : PRINT COMMON RECORD 42 HEADER.
*          : GET ADDRESSABILITY TO THE SUBTYPE DATA FROM THE SUBTYPE
*          : HEADER.
* INPUT    : R9 - SMF 42 RECORD
* OUTPUT   : BUFFER - RECORD 42 HEADER OUTPUT LINE PRINTED
*          : SUBT78@ - POINTER TO THE SUBTYPE DATA PORTION
*          : SUBTOFF - OFFSET TO THE SUBTYPE DATA
*          : SUBTNUM - TOTAL NUMBER OF SUBTYPES TO PROCESS
*****
HHEAD   DS  0H
        ST   R7,RETADDR      SAVE RETURN ADDRESS
*
        PUT  OUTPDD,NULLREC
        PUT  OUTPDD,LSMF42H   INTRO
        PUT  OUTPDD,LSMF421   LABELS FOR HEADER 42
        PUT  OUTPDD,LSMF422
        PUT  OUTPDD,LSMF423
        MVI  BUFFER,X'40'     CLEAR OUTPUT BUFFER
        MVC  BUFFER+1(BUFFLEN-1),BUFFER
*
        MVC  INNDATA(1),SMF42FLG SYSTEM FLAGS
        LA   R10,1
        BAL  R7,HEX2CHAR
        MVC  042FLG(2),OUTDATA
*
        SLR  R4,R4
        IC   R4,SMF42RTY      RECORD TYPE
        CVD  R4,PKD
        UNPK 042TYPE(2),PKD
        MVZ  042TYPE+1(1),042TYPE+0

```

```

*
ICM R4,15,SMF42TME      TIME
ST  R4,INNDATA
LA  R10,4
BAL R7,HEX2CHAR
MVC 042TIME(8),OUTDATA
*
ICM R4,15,SMF42DTE      DATE
ST  R4,INNDATA
LA  R10,4
BAL R7,HEX2CHAR
MVC 042DATE(7),OUTDATA
*
MVC 042SYSID(4),SMF42SID SYSTEM ID
MVC 042SSID(4),SMF42SSI SUBSYSTEM ID
*
SLR R4,R4
ICM R4,3,SMF42STY      SUBTYPE
CVD R4,PKD
UNPK 042STYP(4),PKD
MVZ 042STYP+3(1),042STYP+2
*
LR  R4,R9
A   R4,SMF420PS
USING SMF42PRD,R4
MVC 042PDLV(8),SMF42PDL PRODUCT LEVEL
MVC 042PDNM(10),SMF42PDN PRODUCT NAME
DROP R4
*
* SUBTYPE HEADER
*
LR  R6,R9
LA  R6,SMF42LEN(R6)    POINT TO END OF HEADER 42
USING SMF427H,R6      POINT TO SUBTYPE HEADER
*
L   R4,SMF42NFO        OFFSET TO SUBTYPE SECTION
ST  R4,SUBTOFF         SAVE SECTION OFFSET
AR  R4,R9              POINT TO BEGINING OF SUBTYPE SECTION
ST  R4,SUBT78@
*
SLR R4,R4
ICM R4,3,SMF42NFL      LENGTH OF SUBTYPE SECTION
ST  R4,SUBTLEN         SAVE SECTION LENGTH
*
ICM R4,3,SMF42NFN      NUMBER OF SUBTYPE SECTIONS
ST  R4,SUBTNUM         SAVE NUMBER OF SECTIONS
CVD R4,PKD
UNPK 042STPN(4),PKD
MVZ 042STPN+3(1),042STPN+2
DROP R6

```

```

*
      PUT  OUTPDD,BUFFER      WRITE HEADER 42 INFO
      PUT  OUTPDD,NULLREC
      L    R7,RETADDR        GET RETURN ADDRESS
      BR   R7                RETURN
      EJECT
*****
* SUBROUTINE: ST7
* FUNCTION  : PRINT SUBTYPE 7 INFORMATION.
* INPUT    : SUBT78@ - POINTER TO THE SUBTYPE DATA PORTION
*          : SUBTOFF - OFFSET TO THE SUBTYPE DATA
*          : SUBTNUM - TOTAL NUMBER OF SUBTYPES TO PROCESS
* OUTPUT   : BUFFER - SUBTYPE 7 OUTPUT LINE(S) PRINTED
*****
      ST7      DS 0H
              ST  R7,RETADDR      SAVE RETURN ADDRESS
              L   R3,SUBTNUM      GET NUMBER OF SECTIONS TO PROCESS
              L   R6,SUBT78@      GET SUBTYPE SECTION ADDRESS
              USING SMF42S7,R6
      ST7P    LTR  R3,R3          ANY MORE
              BZ  ST7DONE
              PUT  OUTPDD,LSMF427H  SUBTYPE 7 HEADER
              PUT  OUTPDD,LSMF4271  SUBTYPE LABELS
              PUT  OUTPDD,LSMF4272
              PUT  OUTPDD,LSMF4273
*
      MVI  BUFFER,X'40'        CLEAR OUTPUT BUFFER
      MVC  BUFFER+1(BUFFLEN-1),BUFFER
*
      SLR  R4,R4
      IC   R4,SMF42FFS          FILESYSTEM TYPE
      CVD  R4,PKD
      UNPK ST7FST(2),PKD
      MVZ  ST7FST+1(1),ST7FST
*
      IC   R4,SMF42FTY          FILES TYPE
      CVD  R4,PKD
      UNPK ST7FT(2),PKD
      MVZ  ST7FT+1(1),ST7FT
*
      IC   R4,SMF42FTM          MVS DATASET TYPE
      CVD  R4,PKD
      UNPK ST7MVS(2),PKD
      MVZ  ST7MVS+1(1),ST7MVS
*
      L    R4,SMF42FSN          FILE SERIAL
      CVD  R4,PKD
      UNPK ST7FLS(8),PKD
      MVZ  ST7FLS+7(1),ST7FLS+6
*

```

```

L    R4,SMF42FDN      DEVICE NUMBER FOR FILE
CVD  R4,PKD
UNPK ST7DVT(8),PKD
MVZ  ST7DVT+7(1),ST7DVT+6
*
L    R4,SMF42FIR      NUMBER OF I/O BLOCKS READ
CVD  R4,PKD
UNPK ST7IBR(8),PKD
MVZ  ST7IBR+7(1),ST7IBR+6
*
L    R4,SMF42FIW      NUMBER OF I/O BLOCKS WRITTEN
CVD  R4,PKD
UNPK ST7IBW(8),PKD
MVZ  ST7IBW+7(1),ST7IBW+6
*
* IF OVERFLOW OCCURS WHILE CONVERTING A FLOATING POINT TO DECIMAL,
* THEN OUTPUT WILL BE PRINTED AS IS (HEX-16 BYTES).
LD   FPR0,SMF42FBR    TOTAL BYTES READ CONVERSION
BAL  R7,FP2HEX
LTR  R15,R15          TEST FOR OVERFLOW
BNZ  ST7OFLW1
CVD  R4,PKD
UNPK ST7TRD(8),PKD
MVZ  ST7TRD+7(1),ST7TRD+6
B    ST7CONT1
ST7OFLW1 DS  0H          OVERFLOWED - PRINT OUT AS IS
MVC  INNDATA(8),SMF42FBR TOTAL BYTES READ - AS IS
LA   R10,8
BAL  R7,HEX2CHAR
MVC  ST7TRD(16),OUTDATA
* IF OVERFLOW OCCURS WHILE CONVERTING A FLOATING POINT TO DECIMAL,
* THEN OUTPUT WILL BE PRINTED AS IS (HEX-16 BYTES).
ST7CONT1 DS  0H
LD   FPR0,SMF42FBW    TOTAL BYTES WRITTEN CONVERSION
BAL  R7,FP2HEX
LTR  R15,R15          TEST FOR OVERFLOW
BNZ  ST7OFLW2
CVD  R4,PKD
UNPK ST7TRW(8),PKD
MVZ  ST7TRW+7(1),ST7TRW+6
B    ST7CONT2
ST7OFLW2 DS  0H          OVERFLOWED - PRINT OUT AS IS
MVC  INNDATA(8),SMF42FBW TOTAL BYTES WRITTEN
LA   R10,8
BAL  R7,HEX2CHAR
MVC  ST7TRW(16),OUTDATA
*
ST7CONT2 DS  0H
SLR  R5,R5
ICM  R5,3,SMF42FNL    LENGTH OF FILE NAME

```

```

*
      C      R5,=X'44'          DO NOT EXCEED 44 CHARS
      BNH    ST7CONT3
      LA     R5,44              RESET TO 44 CHARS
ST7CONT3 LR     R3,R5
      LA     R2,ST7FN
      LA     R4,SMF42FFN
      MVCL   R2,R4              FILE NAME
*
      PUT    OUTPDD,BUFFER      SUBTYPE 7 LINE
      PUT    OUTPDD,NULLREC
      LR     R2,R6
      A      R2,SMF42FLO        CALCULATE ADDR OF CLIENT ID SECTION
      BAL    R7,CLID
      DROP   R6
      L      R3,SUBTNUM
      BCTR   R3,R0              DECREMENT COUNTER
      ST     R3,SUBTNUM        SAVE COUNTER
      A      R6,SUBTLEN        NEXT RECORD
ST7DONE  B      ST7P
      DS     0H
      L      R7,RETADDR        GET RETURN ADDRESS
      BR     R7                RETURN
      EJECT
*****
* SUBROUTINE: ST8
* FUNCTION   : PRINT SUBTYPE 8 INFORMATION.
* INPUT      : SUBT78@ - POINTER TO THE SUBTYPE DATA PORTION
*             SUBTOFF - OFFSET TO THE SUBTYPE DATA
*             SUBTNUM - TOTAL NUMBER OF SUBTYPES TO PROCESS
* OUTPUT     : BUFFER - SUBTYPE 8 OUTPUT LINE(S) PRINTED
*****
ST8      DS 0H
      ST     R7,RETADDR        SAVE RETURN ADDRESS
*
      PUT    OUTPDD,LSMF428H    SUBTYPE 8 HEADER
      PUT    OUTPDD,LSMF4281    SUBTYPE LABELS
      PUT    OUTPDD,LSMF4282
      PUT    OUTPDD,LSMF4283
      L      R6,SUBT78@        GET SUBTYPE SECTION ADDRESS
      USING  SMF42S8,R6
      L      R3,SUBTNUM        GET NUMBER OF SECTIONS TO PROCESS
ST8P     LTR   R3,R3            ANY MORE
      BZ     ST8DONE
*
      MVI    BUFFER,X'40'      CLEAR OUTPUT BUFFER
      MVC    BUFFER+1(BUFFLEN-1),BUFFER
*
      STCKCONV STCKVAL=SMF42UST,CONVVAL=OUTDATA,TIMETYPE=DEC,
              DATETYPE=MDDYYYY
      MVC    INNDATA(8),OUTDATA GET START SESSION TIME
      MVC    PKD(4),OUTDATA+8   SAVE START SESSION DATE

```

```

LA    R10,8
BAL   R7,HEX2CHAR
MVC   ST8STAT(6),OUTDATA
MVC   INNDATA(4),PKD      GET START SESSION DATE
LA    R10,4
BAL   R7,HEX2CHAR
MVC   ST8STAD(8),OUTDATA
STCKCONV   STCKVAL=SMF42UET,CONVVAL=OUTDATA,TIMETYPE=DEC,
           DATETYPE=MDDYYYY
MVC   INNDATA(8),OUTDATA  GET STOP SESSION TIME
MVC   PKD(4),OUTDATA+8    SAVE STOP SESSION DATE
LA    R10,8
BAL   R7,HEX2CHAR
MVC   ST8STOT(6),OUTDATA
MVC   INNDATA(4),PKD      GET START SESSION DATE
LA    R10,4
BAL   R7,HEX2CHAR
MVC   ST8STOD(8),OUTDATA
*
MVC   INNDATA(4),SMF42UEL  SESSION ELAPSED TIME
LA    R10,4
BAL   R7,HEX2CHAR
MVC   ST8SEL(8),OUTDATA
*
L     R4,SMF42UNR          TOTAL RPC REQUEST
CVD   R4,PKD
UNPK  ST8TRPC(8),PKD
MVZ   ST8TRPC+7(1),ST8TRPC+6
*
MVC   INNDATA(4),SMF42UTE  TOTAL ELAPSED TIME OF RPC REQUESTS
LA    R10,4
BAL   R7,HEX2CHAR
MVC   ST8TRPE(8),OUTDATA
*
MVC   INNDATA(4),SMF42UAT  TOTAL ACTIVE TIME OF RPC REQUESTS
LA    R10,4
BAL   R7,HEX2CHAR
MVC   ST8TRPA(8),OUTDATA
*
* IF OVERFLOW OCCURS WHILE CONVERTING A FLOATING POINT TO FIXED,
* THEN OUTPUT WILL BE PRINTED AS IS (HEX-16 BYTES).
LD    FPR0,SMF42URN        TOTAL BYTES READ IN FROM NETWORK
BAL   R7,FP2HEX
LTR   R15,R15              TEST FOR OVERFLOW
BNZ   ST8OFLW1
CVD   R4,PKD
UNPK  ST8TNWR(8),PKD
MVZ   ST8TNWR+7(1),ST8TNWR+6
B     ST8CONT1

```

```

ST80FLW1 DS    0H                OVERFLOWED - PRINT OUT AS IS
          MVC  INNDATA(8),SMF42URN TOTAL BYTES READ IN FROM NETWORK
          LA   R10,8
          BAL  R7,HEX2CHAR
          MVC  ST8TNWR(16),OUTDATA
*
  IF OVERFLOW OCCURS WHILE CONVERTING A FLOATING POINT TO FIXED,
* THEN OUTPUT WILL BE PRINTED AS IS (HEX-16 BYTES).
ST8CONT1 DS    0H
          LD   FPR0,SMF42UWN      TOTAL BYTES WRITTEN OUT TO NETWORK
          BAL  R7,FP2HEX
          LTR  R15,R15            TEST FOR OVERFLOW
          BNZ  ST80FLW2
          CVD  R4,PKD
          UNPK ST8TNWW(8),PKD
          MVZ  ST8TNWW+7(1),ST8TNWW+6
          B    ST8CONT2
ST80FLW2 DS    0H                OVERFLOWED - PRINT OUT AS IS
          MVC  INNDATA(8),SMF42UWN TOTAL BYTES WRITTEN OUT TO NETWORK
          LA   R10,8
          BAL  R7,HEX2CHAR
          MVC  ST8TNWW(16),OUTDATA
*
ST8CONT2 DS    0H
*
* IF OVERFLOW OCCURS WHILE CONVERTING A FLOATING POINT TO FIXED,
* THEN OUTPUT WILL BE PRINTED AS IS (HEX-16 BYTES).
          LD   FPR0,SMF42URF      TOTAL BYTES READ FROM FILES
          BAL  R7,FP2HEX
          LTR  R15,R15            TEST FOR OVERFLOW
          BNZ  ST80FLW3
          CVD  R4,PKD
          UNPK ST8TBR(8),PKD
          MVZ  ST8TBR+7(1),ST8TBR+6
          B    ST8CONT3
ST80FLW3 DS    0H                OVERFLOWED - PRINT OUT AS IS
          MVC  INNDATA(8),SMF42URF TOTAL BYTES READ FROM FILES
          LA   R10,8
          BAL  R7,HEX2CHAR
          MVC  ST8TBR(16),OUTDATA
*
* IF OVERFLOW OCCURS WHILE CONVERTING A FLOATING POINT TO FIXED,
* THEN OUTPUT WILL BE PRINTED AS IS (HEX-16 BYTES).
ST8CONT3 DS    0H
          LD   FPR0,SMF42UWF      TOTAL BYTES WRITTEN TO FILES
          BAL  R7,FP2HEX
          LTR  R15,R15            TEST FOR OVERFLOW
          BNZ  ST80FLW4
          CVD  R4,PKD
          UNPK ST8TBW(8),PKD
          MVZ  ST8TBW+7(1),ST8TBW+6
          B    ST8CONT4

```

```

ST80FLW4 DS  0H                OVERFLOWED - PRINT OUT AS IS
          MVC  INNDATA(8),SMF42UWF TOTAL BYTES WRITTEN TO FILES
          LA   R10,8
          BAL  R7,HEX2CHAR
          MVC  ST8TBW(16),OUTDATA
*
ST8CONT4 DS  0H
          PUT  OUTPDD,BUFFER      SUBTYPE 8 LINE
          PUT  OUTPDD,NULLREC
          LR   R2,R9
          A    R2,SMF42UCO        CALCULATE ADDR OF CLIENT ID SECTION
          BAL  R7,CLID
          DROP R6
          L    R3,SUBTNUM
          BCTR R3,R0              DECREMENT COUNTER
          ST   R3,SUBTNUM        SAVE COUNTER
          A    R6,SUBTLEN        NEXT RECORD
          B    ST8P
ST8DONE  DS  0H
          L    R7,RETADDR        GET RETURN ADDRESS
          BR   R7                RETURN

```

```

* SUBROUTINE: CLID
* FUNCTION   : PRINT CLIENT ID INFORMATION.
* INPUT      : R2 - CLIENT ID POINTER
* OUTPUT     : BUFFER - CLIENT ID OUTPUT LINE PRINTED
*****

```

```

CLID     DS 0H
          PUT  OUTPDD,LSMFCLH    CLIENT ID HEADER
          PUT  OUTPDD,LSMFCL1    CID LABELS
          PUT  OUTPDD,LSMFCL3
*
          MVI  BUFFER,X'40'      CLEAR OUTPUT BUFFER
          MVC  BUFFER+1(BUFFLEN-1),BUFFER
*
          USING SMF42CS,R2
          MVC  CLMVSU(8),SMF42CRI MVS USERID
          MVC  CLMVSG(8),SMF42CRG MVS GROUP ID
          MVC  CLMVSA(8),SMF42CAN ACCOUNT GROUP

          L    R4,SMF42CUI      UID
          CVD  R4,PKD
          UNPK CLUID(8),PKD
          MVZ  CLUID+7(1),CLUID+6
*
          L    R4,SMF42CGI      UID
          CVD  R4,PKD
          UNPK CLGID(8),PKD
          MVZ  CLGID+7(1),CLGID+6
*

```

```

SLR  R4,R4
IC   R4,SMF42CIP      IP @ = XXX
CVD  R4,PKD
UNPK CLIP0(3),PKD
MVZ  CLIP0+2(1),CLIP0+1
IC   R4,SMF42CIP+1    IP @ = XXX.XXX
CVD  R4,PKD
UNPK CLIP1(3),PKD
MVZ  CLIP1+2(1),CLIP1+1
IC   R4,SMF42CIP+2    IP @ = XXX.XXX.XXX
CVD  R4,PKD
UNPK CLIP2(3),PKD
MVZ  CLIP2+2(1),CLIP2+1
IC   R4,SMF42CIP+3    IP @ = XXX.XXX.XXX.XXX
CVD  R4,PKD
UNPK CLIP3(3),PKD
MVZ  CLIP3+2(1),CLIP3+1
*
SLR  R5,R5
ICM  R5,3,SMF42CHL    HOSTNAME LENGTH
*
LR   R3,R5
LA   R4,SMF42CHN
LA   R2,CLHN
DROP R2
MVCL R2,R4
*
PUT  OUTPDD,BUFFER    SUBTYPE 7 LINE
PUT  OUTPDD,NULLREC
BR   R7               RETURN
*****
* SUBROUTINE: FP2HEX
* FUNCTION  : CONVERTING FLOATING POINT TO FIXED
* INPUT    : FPR0 - FLOATING POINT NUMBER (DOUBLE WORD)
* OUTPUT   : R4   - CONVERTED VALUE      (FULL WORD)
*          : R15  - RETURN CODE
*          :      0, OK
*          :      8, IF OVERFLOWED
*****
FP2HEX DS 0H
SLR  R15,R15          SET RC = 0
SD   FPR0,TWO31R
BC   11,FP2HEX1
AW   FPR0,TWO32
BC   4,FP2HEX1
STD  FPR0,PKD
XI   PKD+4,X'80'
L    R4,PKD+4
B    FP2DONE

```

```

FP2HEX1 LA R15,8 OVERFLOW; RC = 8
FP2DONE DS 0H
BR R7 RETURN
*
*****
* SUBROUTINE: HEX2CHAR
* FUNCTION : CONVERT HEX TO CHARACTER FOR DISPLAY
* INPUT : R10 - LENGTH OF INPUT FIELD
* INNDATA - HEX DATA TO BE CONVERTED (8 BYTE FIELD)
* OUTPUT: OUTDATA - CONVERT DATA (16 BYTE FIELD)
*****
HEX2CHAR DS 0H
STM R1,R7,SAVEREGS
*
SLR R1,R1
SLR R2,R2
SLR R4,R4 CLEAR EVEN-ODD PAIR
SLR R5,R5
LA R3,INNDATA INPUT BUFFER
LA R6,OUTDATA OUTPUT BUFFER
LOOP1 IC R4,0(R3) GET NEXT BYTE
SRDL R4,4
STC R4,FUNC
TRT FUNC(1),TABLE
STC R2,0(R6)
SLR R4,R4
SLDL R4,4
STC R4,FUNC
TRT FUNC(1),TABLE
STC R2,1(R6)
LA R3,1(R3)
LA R6,2(R6)
BCT R10,LOOP1
*
LM R1,R7,SAVEREGS
BR R7 RETURN
*
DROP R9 SMF 42 MAPPING
EJECT
*****
* DECLARES
*****
SAVEAREA DS 18F SAVE AREA
LABELEN EQU 200 MAX LENGTH OF LABEL
BUFFLEN EQU 200 MAX LENGTH OF OUTPUT LINE
DS 0D
TWO32 DC XL8'4E00000100000000' USED BY FP2HEX
TWO31R DC XL8'4F00000008000000' USED BY FP2HEX
TABLE DS 0H
DC CL16'0123456789ABCDEF' 1-9 A-Z; USED BY HEX2CHAR
DC XL240'00'

```



```

042PDLV DS CL8
        ORG BUFFER+72
042PDNM DS CL10
        ORG ,
        ORG BUFFER+2
ST7FST DS CL2
        ORG BUFFER+7
ST7FT DS CL2
        ORG BUFFER+12
ST7MVS DS CL2
        ORG BUFFER+18
ST7FLS DS CL8
        ORG BUFFER+27
ST7DVT DS CL8
        ORG BUFFER+36
ST7IBR DS CL8
        ORG BUFFER+45
ST7IBW DS CL8
        ORG BUFFER+54
ST7TRD DS CL16
        ORG BUFFER+71
ST7TRW DS CL16
        ORG BUFFER+88
ST7FN DS CL44
        ORG ,
* SUBTYPE 8 INFO-----
        ORG BUFFER+1
ST8STAT DS CL6
        ORG BUFFER+09
ST8STAD DS CL4
        ORG BUFFER+18
ST8STOT DS CL6
        ORG BUFFER+26
ST8STOD DS CL4
        ORG BUFFER+35
ST8SEL DS CL8
        ORG BUFFER+44
ST8TRPC DS CL8
        ORG BUFFER+53
ST8TRPE DS CL8
        ORG BUFFER+63
ST8TRPA DS CL8
        ORG BUFFER+73
ST8TNWR DS CL16
        ORG BUFFER+90
ST8TNWW DS CL16
        ORG BUFFER+106
ST8TBR DS CL16
        ORG BUFFER+123
ST8TBW DS CL16
        ORG ,

```

```

* CLIENT ID INFO -----
CLMVSU  ORG  BUFFER+30
        DS   CL8
        ORG  BUFFER+39
CLMVSG  DS   CL8
        ORG  BUFFER+48
CLMVSA  DS   CL8
        ORG  BUFFER+57
CLUID   DS   CL8
        ORG  BUFFER+66
CLGID   DS   CL8
        ORG  BUFFER+75
CLIP0   DS   CL3
        ORG  BUFFER+79
CLIP1   DS   CL3
        ORG  BUFFER+83
CLIP2   DS   CL3
        ORG  BUFFER+87
CLIP3   DS   CL3
        ORG  BUFFER+92
CLHN    DS   CL63
        ORG  ,
RECCNT  DS   F
S78CNT  DS   F
SAVEREG DS   F
RETADDR DS   F
SUBT78@ DS   F          START OF SUBTYPE SECTION (7 OR 8)
SUBTOFF DS   F          OFFSET TO A SECTION (7 OR 8)
SUBTLEN DS   F          LENGTH OF A SECTION
SUBTNUM DS   F          NUMBER OF SECTIONS
PKD     DS   D
SAVEREGS DS  CL32
FUNC    DS   CL1
        DS   CL3
INNDATA DS   CL8
OUTDATA DS   CL16
IRECORD DS  CL32767     INPUT RECORD
PARMSZ  EQU  *-PARMLIST
        END

```

Appendix L. SMF C and assembler header macros

The following text contains copies of the SMF C and Assembler header macros for your reference. The SMF C header macro is GFSASSMF and the Assembler header macro is GFS AUSMF. Both header macros can be found in the SYS1.NFSMAC library.

SMF C header macro GFSASSMF

Figure 20 contains information about the SMF C header macro GFSASSMF. The GFSASSMF can be found in the SYS1.NFSMAC library.

```
/*
/*****
/*
/* $MAC(GFSASSMF) COMP(5695DF121): NFSS XI
/*
/* MACRO NAME: GFSASSMF.H (SMF.H)
/*
/* DESCRIPTION: Contains C Language Mapping of SMF type 42
/*               and the subtype 7 and 8 records.
/*
/* STATUS: Version 1 Release 1
/*
/* COPYRIGHT:
/*Proprietary V2 Statement
/*Licensed Materials - Property Of IBM
/*"Restricted Materials of IBM"
/*5695-DF1 (C) COPYRIGHT 1994 IBM CORP.
/*End Proprietary V2 Statement
/*
/* FUNCTION: This header file contains the mapping for SMF
/*            records in the following format:
/*
/*            -----
/*            | Type 42 Header |
/*            -----
/*            | Subtype 7 or 8 Header |
/*            -----
/*            | Product Section |
/*            -----
/*            | Subtype 7 or 8 Data |
/*            -----
/*            | Client Section |
/*            -----
/*
/* Note: 1) To obtain address of the subtype 7 or 8 header
/*         add SMF42LEN to the address of the type 42 header
/*         2) To obtain the address of the product section
/*
```

Figure 20. SMF C header macro GFSASSMF

```

/*      add SMF420PS to the address of the type 42 header
/*      3) a) To obtain the address of the subtype 7 data portion
/*           add SMF42NFO to the address of the type 42 header
/*           b) To obtain the address of the subtype 8 data portion
/*           add SMF42NU0 to the address of the type 42 header
/*      4) a) To obtain the address of the subtype 7 client
/*           section add SMF42FL0 to the address of the
/*           current subtype 7 data portion. There can be more
/*           than one subtype 7 record in a type 42 record.
/*           There will always be a subtype 7 and client pair.
/*           b) To obtain the address of the subtype 8 client
/*           section add SMF42UC0 to the address of the
/*           type 42 header.
/*
/*
/* NOTES:
/*   DEPENDENCIES: All changes made to this macro must be
/*                  reflected in GFSASMF. If any of the header
/*                  sections change, they will also have to be
/*                  reflected in IGWSMF
/*
/*   MODULE TYPE:  C header/include file
/*   PROCESSOR:    IBM ADC/370 compiler
/*   LIBRARY:      NFSLB
/*   CC OPTIONS:   None.
/*   ATTRIBUTES:   None.
/*
/*   LINKAGE: #include "GFSASsmf.h"
/*
/*
/* EXTERNAL REFERENCES: (external to this header file)
/*   DATA AREAS: None
/*   INCLUDE FILES: None
/*
/* TYPEDEFS DEFINED:
/*   smf42
/*   smf427h
/*   smf428h
/*   smf42prd
/*   smf42s7
/*   smf42cs
/*   smf42s8
/*
/* STRUCTs DEFINED:
/*   None
/*
/* MACROs DEFINED:
/*   None
/*
/******

```

```

/*-----*/
/* Header for SMF record type 42          */
/*-----*/

typedef _Packed struct smf42 {
short int    smf42rc1;    /* Record Length
short int    smf42sgd;    /* Segment Descriptor (RDW) -- 0 if
                        /* record is not spanned
unsigned char smf42flg;    /* System indicator flags
char         smf42rty;    /* Record type: 42 (X'2A')
int         smf42tme;    /* Time in hundredths of a second
unsigned char smf42dte[4]; /* Date record written (by SMF)
unsigned char smf42sid[4]; /* System identification (by SMF)
unsigned char smf42ssi[4]; /* Subsystem Id
short int    smf42sty;    /* Record subtype
short int    smf42nt;    /* Number of triplets (optional)
short int    _filler1;    /* Reserved (optional)

/*-----*/
/* Product section triplet                */
/*-----*/

int         smf42ops;    /* Offset to product section
short int   smf42lps;    /* Length of product section
short int   smf42nps;    /* Number of product sections

/*-----*/
/* Header must end on word boundary      */
/*-----*/
/* unsigned char smf42end;    1st data section triplet
} SMF42;

/* Values for field "smf42flg"
#define smf42fsi 0x80 /* When set=subsystem id follows system id
#define smf42fsu 0x40 /* When set = subtypes are used
#define smf42fxa 0x04 /* When set = MVS/XA (SMF enters)
#define smf42fs2 0x02 /* When set = VS2 (SMF enters)
#define smf42fs1 0x01 /* When set = VS1 (SMF enters)

#define smf42_len 0x28

/*-----*/
/* SMF42 subtype 7 header section
/* (file timeout statistics)
/*-----*/

```

```

typedef struct smf427h {
    int          smf42nfo; /*Offset NFSS file timeout stats section
    short int    smf42nfl; /*Length NFSS file timeout stats section
    short int    smf42nfn; /*Number NFSS file timeout stats section
} SMF427H;

```

```

/*-----
/* SMF42 subtype 8 header section
/* (user logout statistics )
/*-----

```

```

typedef struct smf428h {
    int          smf42nuo; /* Offset NFSS user session stats section
    short int    smf42nul; /* Length NFSS user session stats section
    short int    smf42nun; /* Number NFSS user session stats section
} SMF428H;

```

```

/*-----*/
/* Product Section                               */
/*-----*/

```

```

typedef struct smf42prd {
    char          smf42pd1[8]; /* Product Level
    char          smf42pdn[10]; /* Product Name
    char          filler[22]; /* Reserved
} SMF42PRD;

```

```

/*----- */
/* SMF type 42 subtype 7 file timeout Statistics */
/*----- */

```

```

typedef _Packed struct smf42s7 {
    int          smf42flo; /*Offset to client identification section.
    short int    smf42fl1; /*Length of client identification section.
    char          smf42ffs; /* File system type indicator.
                        /* 01 = HFS
                        /* 02 = MVS
    char          smf42fty; /* File type as defined in NFS protocol.
                        /* 0 = Non-file
                        /* 1 = Regular file
                        /* 2 = Directory
                        /* 3 = Block special device
                        /* 4 = Character special device
                        /* 5 = Symbolic link
    char          smf42ftm; /* MVS data set type.
                        /* 0 = unknown MVS file type
                        /* 1 = Sequential (BSAM) file
                        /* 2 = Partitioned (BPAM)
                        /* 3 = Direct Access file
                        /* 4 = ISAM is not supported

```

```

        /* 5 = Virtual Sequential Access
        /* 6 = VSAM Entry Sequenced
        /* 7 = VSAM Relative Record
        /* 8 = VSAM Keyed access
        /* 9 = dummy index level file block
        /* 10 = HFS file type
char      s7fill1[3]; /* Reserved
int       smf42fsn; /* File Serial Number, HFS INODE #
int       smf42fdn; /* Unique device number, HFS file
           /* system number
int       smf42fir; /* Number of I/O blocks read.
int       smf42fiw; /* Number of I/O blocks written.
int       s7fill2; /* Reserved
double    smf42fbr; /* Number of bytes read from file.
double    smf42fbw; /* Number of bytes written to file.
short int smf42fnl; /* Length of file name.
char      s7fill3[6]; /* Reserved - Dword boundary
/*-----*/
/* This is followed by the file name. It is      */
/* either a 256 byte character string            */
/* (SMF_SHORT_FNAME) or a 1023 byte character    */
/* string (SMF_LONG_FNAME). See the S7RECSHORT  */
/* and S7RECLONG typedefs below.                */
/*-----*/
} SMF42S7;

/*-----*/
/* Client Identification Section                  */
/*-----*/

typedef struct smf42cs {
char      smf42cri[8]; /* RACF user ID.
char      smf42crg[8]; /* RACF group name.
char      smf42can[8]; /* Account Number.
int       smf42cui; /* User ID at client host (UNIX style)
int       smf42cgi; /* Group ID at client host (UNIX style)
int       smf42cip; /* IP address of client host.
short int smf42chl; /* Length of client host name
char      smf42chn[63]; /* Client host name.
} SMF42CS;

/*-----*/
/* SMF type 42 subtype 8 user session completion Statistics */
/*-----*/

typedef struct smf42s8 {
int       smf42uco; /* Offset client identification section
short int smf42ucl; /* Length client identification section
short int smf42res2; /* Reserved.
double    smf42ust; /* Session start time (in STCK format)
double    smf42uet; /* Session end time (in STCK format)

```

```

int          smf42uel; /* Session elapsed time (milliseconds)
u_long      smf42unr; /* Number of RPC requests processed in
                  /* this session
int          smf42ute; /* Total elapsed time of all RPC
                  /* requests processes in this session.
int          smf42uat; /* Total active time of all RPC
                  /* requests processes in this session.
double      smf42urn; /* Number of bytes read in from the
                  /* network in this session
double      smf42uwn; /* Number of bytes written out to the
                  /* network in this session
double      smf42urf; /* Number of bytes read from files on
                  /* this session
double      smf42uwf; /* Number of bytes written to files in
                  /* this session

} SMF42S8;
/*****
/* the following typedefs are used internally to construct and
/* free SMF type 42 subtype 7 data chaining elements.
/* elements.
/*

#define SMF_SHORT_FNAME 256      /* short file name <= 256
#define SMF_LONG_FNAME 1023     /* long file name > 256
#define SMF_MAX_RECLEN 32390    /* maximum SMF record length
/* Length of file usage stat., plus short file name, plus client data
#define SMF_SHORT_DATALEN \
    (sizeof(SMF42S7)+SMF_SHORT_FNAME+sizeof(SMF42CS))
#define SMF_LONG_DATALEN \
    (sizeof(SMF42S7)+SMF_LONG_FNAME+sizeof(SMF42CS))

/* S7ELEM is gotten whenever a file times out, to collect statistics
   on the file. It is then chained to the list from NFSGLOBAL, and
   the chain is processed later to write SMF dataset usage record
   subtype 7. This is done for files with file name <= 256.
typedef _Packed struct s7elem {      /* element to be chained
_Packed struct s7elem *next;        /* next element in list
    char *filler;                    /* account for C forcing
                                      /* doubleword alignment
    SMF42S7 smf42s7;                 /* data portion
    char smfilen[SMF_SHORT_FNAME]; /* 256 byte file name
    SMF42CS smf42cs;                 /* client section
} S7ELEM;                            /*

/* SMF file usage record for a file with file name <= 256
typedef _Packed struct s7recshort { /* type 42 subtype 7
    SMF42 smf42;                     /* type 42 header
    SMF427H smf427h;                 /* subtype 7 header
    SMF42PRD smf42prd;               /* product section
    SMF42S7 smf42s7;                 /* data portion
    char smfilen[SMF_SHORT_FNAME]; /* 256 byte path name

```

```

        SMF42CS smf42cs;          /* client section
} S7RECSHORT;                   /*

/* SMF file usage record for a file with file name > 256
typedef _Packed struct s7reclong { /* type 42 subtype 7
    SMF42 smf42;                 /* type 42 header
    SMF427H smf427h;           /* subtype 7 header
    SMF42PRD smf42prd;         /* product section
    SMF42S7 smf42s7;           /* data portion
    char smfilen[SMF_LONG_FNAME]; /* 1023 byte path name
    SMF42CS smf42cs;           /* client section
} S7RECLONG;                   /*

/* SMF user session statistics record - subtype 8
typedef _Packed struct s8rec { /* type 42 subtype 8
    SMF42 smf42;                 /* type 42 header
    SMF428H smf428h;           /* subtype 8 header
    SMF42PRD smf42prd;         /* product section
    SMF42S8 smf42s8;           /* data portion
    SMF42CS smf42cs;           /* client section
} S8REC;

```

SMF assembler header macro GFS AUSMF

Figure 21 contains information about the SMF Assembler header macro GFS AUSMF. The GFS AUSMF can be found in the SYS1.NFSMAC library.

```

/******
**
** $MAC(GFS AUSMF) COMP(5695DF121): NFSS XI
**
** MACRO NAME: GFS AUSMF.ASM
**
** DESCRIPTION: Contains ASM Language Mapping of SMF type 42
**              and the subtype 7 and 8 records.
**
** STATUS: Version 1 Release 1
**
** COPYRIGHT:
**Proprietary V2 Statement

```

Figure 21. SMF assembler header macro GFS AUSMF

```

/**Licensed Materials - Property Of IBM
**"Restricted Materials of IBM"
**5695-DF1 (C) COPYRIGHT 1994 IBM CORP.
**End Proprietary V2 Statement
**
** FUNCTION: Contains ASM Language Mapping of SMF type 42
**           and the subtype 7 and 8 records in the following
**           format:
**
**           -----
**           | Type 42 Header                               |
**           -----
**           | Subtype 7 or 8 Header                       |
**           -----
**           | Product Section                             |
**           -----
**           | Subtype 7 or 8 Data                         |
**           -----
**           | Client Section                              |
**           -----
**
** Note: 1) To obtain address of the subtype 7 or 8 header
**         add SMF42LEN to the address of the type 42 header
**       2) To obtain the address of the product section
**         add SMF420PS to the address of the type 42 header
**       3) a) To obtain the address of the subtype 7 data portion
**         add SMF42NF0 to the address of the type 42 header
**         b) To obtain the address of the subtype 8 data portion
**         add SMF42NU0 to the address of the type 42 header
**       4) a) To obtain the address of the subtype 7 client
**         section add SMF42FL0 to the address of the
**         current subtype 7 data portion. There can be more
**         than one subtype 7 record in a type 42 record.
**         There will always be a subtype 7 and client pair.
**         b) To obtain the address of the subtype 8 client
**         section add SMF42UC0 to the address of the
**         type 42 header.
**
** NOTES:
** DEPENDENCIES: All changes made to this macro must be
**               reflected in GFSASSMF. If changes are made
**               to any of the header sections, the change
**               will also have to be made in IGWSMF.
**
** MACRO:
** PROCESSOR:   High Level Assembler
** LIBRARY:     NFSLB
** ATTRIBUTES:  Include
**
** EXTERNAL REFERENCES: (external to this header file)

```

```

**/ DATA AREAS:      None
**/ CONTROL BLOCKS:  None
**/ MACROS:          None
**/*****
      MACRO
      GFSASMF
*
*
*****
*   Header for SMF record type 42   *
*****
*
*
SMF42  DSECT
SMF42BAS DS  0D      SMF42BAS is the basing expr.
SMF42RCL DS  H      Record Length
SMF42SGD DS  H      Segment Descriptor (RDW) -- 0 if record is
*                    not spanned
SMF42FLG DS  0BL1   System indicator flags
SMF42FSI EQU  X'80'  When set=subsystem id follows system id
SMF42FSU EQU  X'40'  When set = subtypes are used
SMF42FXA EQU  X'04'  When set = MVS/XA (SMF enters)
SMF42FS2 EQU  X'02'  When set = VS2 (SMF enters)
SMF42FS1 EQU  X'01'  When set = VS1 (SMF enters)
      ORG  SMF42FLG+X'00000001'
SMF42RTY DS  X      Record type: 42 (X'2A')
SMF42TME DS  FL4    Time in hundredths of a second when record
*                    was moved to SMF buffer
SMF42DTE DS  CL4    Date record written (by SMF)
SMF42SID DS  CL4    System identification (by SMF)
SMF42SSI DS  CL4    Subsystem Id
SMF42STY DS  HL2    Record subtype
SMF42NT  DS  HL2    Number of triplets (optional)
      DS  HL2    Reserved (optional)
*
*****
*   Product section triplet   *
*****
*
SMF420PS DS  FL4    Offset to product section
SMF42LPS DS  HL2    Length of product section
SMF42NPS DS  HL2    Number of product sections
*
*****
      Header must end on word boundary   *

```

```

*****
*
SMF42END DS    0F          1st data section triplet
SMF42LEN EQU  *-SMF42
*
*
*****
*      Product Section                                     *
*****
*
SMF42PRD DSECT
SMF42PDL DS    CL8          Product Level
SMF42PDN DS    CL10         Product Name
          DS    CL22         Reserved
*
*****
*      Product section must end on word boundary         *
*****
*
SMF42PEN DS    0F
SMF42PLN EQU  *-SMF42PRD Length of product section
*
*
*****
* SMF42 subtype 7 header section                         *
* (file timeout statistics)                             *
*****
*
SMF427H DSECT
SMF42NF0 DS    F            Offset to NFSS file timeout stats section
SMF42NFL DS    H            Length of NFSS file timeout stats section
SMF42NFN DS    H            Number of NFSS file timeout stats section
SMF427HE EQU  *-SMF427H Length of subtype 7 header
*
*
*****
* SMF42 subtype 8 header section                         *
* (user logout statistics )                             *
*****
*
SMF428H DSECT
SMF42NU0 DS    F            Offset to NFSS user session stats section
SMF42NUL DS    H            Length of NFSS user session stats section
SMF42NUN DS    H            Number of NFSS user session stats section
SMF428HE EQU  *-SMF428H Length of subtype 8 header
*
*
*****
*      SMF type 42 subtype 7 file timeout Statistics     *
*      (file timeout statistics)                         *
*****
*

```

```

SMF42S7 DSECT
SMF42FLO DS F Offset to client identification section
SMF42FLL DS H Length of client identification section
SMF42FFS DS X File system type indicator
*           01 = HFS
*           02 = MVS
SMF42FTY DS X File type as defined in NFS protocol
*           00 = Non-file
*           01 = Regular file
*           02 = Directory
*           03 = Block special device
*           04 = Character special device
*           05 = Symbolic link
SMF42FTM DS X MVS data set type
*           0 = unknown MVS file type
*           1 = Sequential (BSAM) file
*           2 = Partitioned (BPAM)
*           3 = Direct Access file
*           4 = ISAM is not supported
*           5 = Virtual Sequential Access
*           6 = VSAM Entry Sequenced
*           7 = VSAM Relative Record
*           8 = VSAM Keyed access
*           9 = dummy index level file block
*           10 = HFS file type
          DS XL3 Reserved
SMF42FSN DS F File Serial Number, HFS INODE #
SMF42FDN DS F Unique device number HFS file system #
SMF42FIR DS F Number of I/O blocks read
SMF42FIW DS F Number of I/O blocks written
          DS F Reserved
SMF42FBR DS D Number of bytes read from file
SMF42FBW DS D Number of bytes written to file
SMF42FNL DS H Length of file name
SMF42FCL DS 0D C370 ends structure on doubleword boundary
SMF42FFN DS 0D File name
*
* The file name is either a 256 byte character string, or a
* 1023 byte character string. SMF42FNL will contain the length
*
*           Start of Client Section
SMF42F7E EQU *-SMF42S7 Length of subtype 7 data section
*
SMFSNAME EQU 256 Short file name
SMFLNAME EQU 1023 Long file name
*
*
*****
* Client Identification Section *
*****
*
SMF42CS DSECT

```

```

SMF42CRI DS    CL8      RACF user ID
SMF42CRG DS    CL8      RACF group name
SMF42CAN DS    CL8      Account Number
SMF42CUI DS    F        User ID at client host (UNIX style)
SMF42CGI DS    F        Group ID at client host (UNIX style)
SMF42CIP DS    F        IP address of client host
SMF42CHL DS    H        Length of client host name
SMF42CHN DS    CL63     Client host name
                   DS    0F        Round to fullword alignment
SMF42CSE EQU   *-SMF42CS Length of Client Section
*
*
*****
* SMF type 42 subtype 8 user session completion Statistics      *
* (user logout statistics )                                     *
*****
*
SMF42S8 DSECT
SMF42UC0 DS    F        Offset to client identification section
SMF42UCL DS    H        Length of client identification section
                   DS    H        Reserved
SMF42UST DS    D        Session start time (in STCK format)
SMF42UET DS    D        Session end time (in STCK format)
SMF42UEL DS    F        Session elapsed time (in milliseconds)
SMF42UNR DS    F        Number of RPC requests processed in this
*                          session
SMF42UTE DS    F        Total elapsed time of all RPC requests
*                          processes in this session
SMF42UAT DS    F        Total active time of all RPC requests
*                          processes in this session
SMF42URN DS    D        Number of bytes read in from the network in
*                          this session
SMF42UWN DS    D        Number of bytes written out to the network
*                          in this session. */
SMF42URF DS    D        Number of bytes read from files on this
*                          session */
SMF42UWF DS    D        Number of bytes written to files in this
*                          session
SMF42UCS DS    0F        Start of Client Section
SMF42S8E EQU   *-SMF42S8 Length of subtype 8 data section
MEND

```

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Information Enabling Requests
Dept. DZWA
5600 Cottle Road
San Jose, CA 95193 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Programming interface information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of z/OS DFSMS.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, or other countries, or both:

AIX	OpenEdition
DFSMS/MVS	OS/2
DFSMSdfp	OS/390
DFSMSdss	RACF
DFSMSHsm	RETAIN
IBM	Resource Link
IBMLink	RISC System/6000
Language Environment	RS/6000
MVS	Unicode
MVS/DFP	VTAM
MVS/ESA	z/OS

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries

Other company, product, and service names, which may be trademarks or service marks of others.

Glossary

This glossary defines technical terms and abbreviations used in DFSMS documentation. If you do not find the term you are looking for, refer to the index of the appropriate DFSMS manual or view the *IBM Dictionary of Computing Terms* located at:

<http://www.ibm.com/networking/nsg/nsgmain.htm>

This glossary includes terms and definitions from:

- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The *Information Technology Vocabulary* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published part of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.
- The *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.

The following cross-reference is used in this glossary:

See: This refers the reader to (a) a related term, (b) a term that is the expanded form of an abbreviation or acronym, or (c) a synonym or more preferred term.

A

access. To obtain computing services.

access method. (1) A mainframe data management routine that moves data between storage and an I/O device in response to requests made by a program. (2) The part of the distributed data management architecture which accepts commands to access and process the records of a file.

access permission. A group of designations that determine who can access a particular AIX or UNIX file and how the user can access the file.

ACS. See *automatic class selection*.

ACS routine. A procedural set of ACS language statements. Based on a set of input variables, the ACS language statements generate the name of a predefined SMS class, or a list of names of predefined storage groups, for an MVS file.

address. The unique identifier assigned to each device or workstation connected to a network.

address space. The complete range of addresses in memory available to a computer program.

Advanced Interactive Executive (AIX). IBM's licensed version of the UNIX operating system.

AIX. See *Advanced Interactive Executive*.

alias. An alternative name for an ICF user catalog, a non-VSAM file, or a member of a partitioned data set (PDS) or PDSE.

alias entry. An entry that relates an alias to the real entry name of a user catalog or nonVSAM data set.

allocation. (1) Generically, the entire process of obtaining a volume and unit of external storage, and setting aside space on that storage for a data set. (2) The process of connecting a program to a data set or devices.

American Standard Code for Information Interchange (ASCII). The standard code used for information exchange among data processing systems, data communication systems, and associated equipment. It uses a coded character set consisting of 7-bit coded characters.

APAR. See *authorized program analysis report*.

APF. Authorized program facility.

API. See *application programming interface*.

application programming interface (API). A formally defined programming language interface between an IBM system control program or a licensed program and the user of a program.

ASCII. See *American National Standard Code for Information Interchange*.

atime. The time when the file was last accessed.

automatic class selection (ACS). A mechanism for assigning Storage Management Subsystem classes and storage groups to data sets.

automatic class selection (ACS) routine. A procedural set of ACS language statements. Based on a set of input variables, the ACS language statements generate the name of a predefined SMS class, or a list of names of predefined storage groups, for a data set.

authorized program analysis report (APAR). A request for correction of a problem caused by a suspected defect in a current unaltered release of a program.

B

basic sequential access method (BSAM). An access method for storing or retrieving data blocks in a continuous sequence, using either a sequential access or a direct access device.

BIOD. The caching daemon that caches directory lookups and file data when remote files are accessed from the host.

block. A string of data elements recorded, processed, or transmitted as a unit. The elements can be characters, words, or physical records.

BSAM. See *Basic sequential access method*.

C

CCSID. See *Coded Character Set Identifier*.

CDRA. Character data representation architecture.

Character Data Representation Architecture (CDRA)

API. A set of identifiers, services, supporting resources, and conventions for consistent representation, processing, and interchange of character data.

client. (1) A user. (2) A consumer of resources or services. (3) A functional unit that receives shared services from a server. (4) A system that is dependent on a server to provide it with programs or access to programs. (5) On a network, the computer requesting services or data from another computer.

client-server relationship. Any process that provides resources to other processes on a network is a *server*. Any process that employs these resources is a *client*. A machine can run client and server processes at the same time.

Coded Character Set Identifier (CCSID). A 16-bit number that identifies a specific encoding scheme identifier, character set identifiers, code page identifiers,

and additional coding required information. The CCSID uniquely identifies the coded graphic character representation used.

connection. An association established between functional units for conveying information.

current directory. The currently active directory; the directory that is searched when you enter a file name without indicating the directory that contains the file name. When you specify a file name without specifying a directory, the system assumes that the file is in the current directory.

D

daemon. A background process that is initiated at system start that continuously performs a function required by another process.

DA. Direct access.

DAE. Dump Analysis and Elimination.

DAM. Direct access method.

DASD. Direct access storage device.

DASD volume. A Direct Access Storage Device space identified by a common label and accessed by a set of related addresses. See also *volume*, *primary storage*, *migration level 1*, *migration level 2*.

data control block (DCB). A control block used by access method routines in storing retrieving data.

data set. In DFSMS, the major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access. In z/OS non-UNIX environments, the terms *data set* and *file* are generally equivalent and sometimes are used interchangeably. See also *file*. In z/OS UNIX environments, the terms *data set* and *file* have quite distinct meanings.

data set organization. The type of arrangement of data in a data set. Examples are sequential organization or partitioned organization.

DBCS. Double byte character set.

DCB. See *data control block*.

DES authentication. Requires a client to send credentials (its name, conversation key, window key, and a time stamp) to the server. The server then returns a verifier to the client. Data Encryption Standard (DES) credentials are sometimes called "secure" credentials because they are based on a sender's ability to encrypt data using a common time reference. DES-style credentials require a randomly generated key used to encrypt a common reference time which is then used in

turn to create a *conversation key*. DES-style credentials are less vulnerable to penetration than UNIX-style credentials because of the generated conversation key and the data that is encrypted with the conversation key. Other forms of attack (for example, replaying message traffic) are resisted because of the timestamp in the transmission. The Network File System does not support DES-style credentials.

DFSMSdftp. A DFSMS functional component or base element of z/OS, that provides functions for storage management, data management, program management, device management, and distributed data access.

direct access file. A type of MVS file for storing data on a random access device that is accessed using a record address.

directory. A file that maps the names of other directories and files to their locations.

DOS. Disk operating system.

DSCB . Data set control block.

E

EBCDIC. See *extended binary-coded decimal interchange code*.

end user. A person in a data processing installation who requires the services provided by the computer system.

EOR. End of record.

entry-sequenced data set (ESDS). In VSAM, a data set whose records are loaded without respect to their contents and whose RBAs cannot change. Records are retrieved and stored by addressed access, and new records are added at the end of the data set.

ESDS. See *entry-sequenced data set (ESDS)*.

ESS. Enterprise Storage Server.

ETR. Electronic technical response.

exports data set. An MVS file on the server containing entries for directories that can be exported to Network File System clients. It is used by the server to determine which MVS files and prefixes can be mounted by a client, and to write-protect MVS files on the server.

extended binary-coded decimal interchange code (EBCDIC). A coded character set consisting of 8-bit coded characters.

External Data Representation (XDR). A standard developed by Sun Microsystems, Incorporated for representing data in machine-independent format. XDR

is a vendor independent way of representing the data. By using the XDR standard data representation convention, systems do not have to understand and translate every data format that exists on the network; there is only the one convention. Data is translated into XDR format before it is sent over the network and, at the reception point, is translated into the data convention used there. This means that new computer architectures can be integrated into the network without requiring the updating of translation routines. The new architecture simply includes a routine that translates its data format into XDR format and the new member of the network is ready to go.

Using XDR, data can be accessed or exchanged among machines of various hardware and software architectures without any translation or interpretation problems. Word lengths, byte ordering, and floating point representations appear to be the same to all nodes in the network.

F

FDDI. Fiber Distributed Data Interface (FDDI).

file. A collection of information treated as a unit. In non-UNIX System Services MVS environments, the terms *data set* and *file* are generally equivalent and are sometimes used interchangeably. See also *data set*.

file handle. A file handle is used by the client and server sides of the Network File System to specify a particular file or prefix.

file system. In the z/OS UNIX HFS environment, the collection of files and file management structures on a physical or logical mass storage device, such as a diskette or minidisk. See also *HFS data set*.

File Transfer Protocol (FTP). A TCP/IP protocol used for transferring files to and from foreign hosts. FTP also provides the capability to access directories.

FMID. Function modification identifier.

FUB. File usage block.

foreign host. Any host on the network other than the local host.

free space. Space reserved within the control intervals of a key-sequenced data set for inserting new records into the data set in key sequence or for lengthening records already there; also, whole control intervals reserved in a control area for the same purpose.

FTP. File Transfer Protocol.

G

gateway. A functional unit that interconnects two computer networks with the different network

architectures. A gateway connects networks or systems of different architectures. A bridge interconnects networks or systems with the same or similar architectures.

GID. See *group number (GID)*.

group. (1) With respect to partitioned data sets, a member and the member's aliases that exist in a PDS or PDSE, or in an unloaded PDSE. (2) A collection of users who can share access authorities for protected resources.

group number (GID). A unique number assigned to a group of related users. The group number can often be substituted in commands that take a group name as an argument.

GXB . Global exit block.

H

handle. (1) In the Advanced DOS and OS/2 operating systems, a binary value created by the system that identifies a drive, directory, and file so that the file can be found and opened. (2) In the AIX operating system, the data structure that is a temporary local identifier on an object. Allocating a handle creates it. Binding a handle makes it identify an object at a specific location. (3) In the OS/400 application programming interfaces, a variable that represents an object.

HFS. See *hierarchical file system (HFS)*.

hierarchical file system (HFS) data set. A data set that contains a POSIX-compliant hierarchical file system, which is a collection of files and directories organized in a hierarchical structure, that can be accessed using the UNIX System Services MVS facilities.

host. A computer connected to a network that provides an access method to that network. A host provides end-user services.

I

ICF. See *integrated catalog facility (ICF)*.

ICP. Interconnect Control Program (ICP).

IDCAMS. Integrated catalog access methods services.

integrated catalog facility (ICF). In the Data Facility Product (DFP), a facility that provides for integrated catalog facility catalogs.

Internet. A specific internetwork that includes ARPANET, MILNET, and NSFnet. These networks use the TCP/IP protocol suite. See also *internetwork*.

Internet Protocol (IP). The TCP/IP layer between the higher-level host-to-host protocol and the local network protocols. IP uses local area network protocols to carry packets in the form of diagrams to the next gateway or destination host.

internetwork. A collection of packet-switched networks that are connected by gateways. They work as a single network.

interoperability. The ability of hardware and software from different vendors to communicate on a network.

inter-process communication (IPC). Ways for programs to communicate data to each other and to synchronize their activities. Semaphores, signals, and internal message queues are common methods of inter-process communication.

IP. Internet Protocol.

IPC. See *inter-process communication (IPC)*.

J

JCL. See *job control language*.

Job control language (JCL). A problem-oriented language used to identify the job or describe its requirements to an operating system.

K

key-sequenced data set (KSDS). A VSAM data set whose records are loaded in key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in the data set in key sequence because of free space allocated in the data set. Relative byte addresses of records can change because of control interval or control area splits.

KSDS. See *key-sequenced data set (KSDS)*.

L

LFS. Logical file system.

local host. The computer to which a user's terminal is directly connected.

local network. That portion of a network physically connected to the host without intermediate gateways.

M

management class. A collection of management attributes, defined by the storage administrator, used to control the release of allocated but unused space; to control the retention, migration, and backup of data

sets; to control the retention and backup of aggregate groups, and to control the retention, backup, and class transition of objects.

master catalog. A key-sequenced data set or file with an index containing extensive data set and volume information that VSAM requires to locate data sets or files, to allocate and deallocate storage space, to verify the authorization of a program or operator to gain access to a data set or file, and to accumulate usage statistics for data sets or files.

MBCS. Multiple byte character set.

migration level 1. DFSMSHsm-owned DASD volumes that contain data sets migrated from primary storage volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage*, *migration level 2*.

migration level 2. DFSMSHsm-owned tape or DASD volumes that contain data sets migrated from primary storage volumes or from migration level 1 volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage*, *migration level 1*.

mkdir. An AIX, UNIX, OS/2 and DOS command used to make a new directory.

mount. A host-linked operation which results in a tape cartridge being physically inserted into a tape drive.

mount handle data set. A data set used to store the file handles of mount points.

mount point. A directory established in a workstation or a server local directory that is used during the transparent accessing of a remote file.

mtime. The time of last data modification.

MTU. Maximum transmission unit (MTU).

MVS. Multiple virtual system.

MVS/ESA. Multiple Virtual Storage/Enterprise Systems Architecture. A z/OS operating system environment that supports ESA/390.

MVS/ESA SP. An IBM licensed program used to control the z/OS operating system. MVS/ESA SP together with DFSMS compose the base MVS/ESA operating environment. See also z/OS.

mvslogin. A client command to connect your workstation to MVS.

mvslogout (or mvslogout). A client command to break the connection between your workstation and MVS.

N

network. (1) An arrangement of nodes and connection branches. (2) A configuration of data processing devices and software connected for information interchange.

NFS. See *z/OS Network File System*.

NLM. Network Lock Manager.

NSM. Network Status Manager.

null credentials. Null credentials are usually associated with diskless workstations where it is impossible to obtain identifying information because there is no local (that is, local to the workstation) repository of information.

O

object storage hierarchy. A hierarchy consisting of objects stored in DB2 table spaces on DASD, on optical or tape volumes that reside in a library, and an optical or tape volumes that reside on a shelf. See also *storage hierarchy*.

optical volume. Storage space on an optical disk, identified by a volume label. See also *volume*.

OpenEdition MVS. See *z/OS UNIX System Services*.

OpenEdition MVS file system. See *z/OS UNIX file system*.

optical volume. Storage space on an optical disk, identified by a volume label. See also *volume*.

Operating System/2 (OS/2). An operating system used in IBM PC AT, PS/2, and compatible computers.

OSA2. Open System Adapter 2 (OSA2).

P

partitioned data set (PDS). A data set on direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

partitioned data set extended (PDSE). A system-managed data set that contains an indexed directory and members that are similar to the directory and members of partitioned data sets. A PDSE can be used instead of a partitioned data set.

PDS. See *partitioned data set (PDS)*.

PDS directory. A set of records in a partitioned data set (PDS) used to relate member names to their locations on a DASD volume.

PDSE. See *partitioned data set extended (PDSE)*.

permission code. A three-digit octal code, or a nine-letter alphabetic code, indicating the access permissions. The access permissions are read, write, and execute.

permission field. One of the three-character fields within the permissions column of a directory listing indicating the read, write, and run permissions for the file or directory owner, group, and all others.

PFS. Physical file system.

portable operating system interface (POSIX). Portable operating system interface for computer environments. An IEEE operating system standard, closely related to the UNIX system (software writing).

port. (1) An access point for data entry or exit. (2) A receptacle on a device to which a cable for another device is attached.

Portmapper. A server that converts RPC program numbers into port numbers acceptable to the protocol. This server must be running to make RPC calls.

POSIX . See *portable operating system interface (POSIX)*.

primary storage. A DASD volume available to users for data allocation. The volumes in primary storage are called primary volumes. See also *storage hierarchy*. Contrast with *migration level 1*, *migration level 2*.

protocol. (1) A set of semantic and syntactic rules that determines the behavior of functional units in achieving communication. (2) A specification for the format and relative timing of information exchanged between communicating parties.

PS. Physical sequential.

PTF. Program temporary fix.

Q

QSAM. See *queued sequential access method (QSAM)*.

queued sequential access method (QSAM). An extended version of the basic sequential access method (BSAM). Input data blocks awaiting processing or output data blocks awaiting transfer to auxiliary storage are queued on the system to minimize delays in I/O operations.

R

RACF. See *Resource Access Control Facility (RACF)*.

RBA. Relative byte address.

relative byte address (RBA). The displacement of a data record or a control interval from the beginning of the data set to which it belongs; independent of the manner in which the data set is stored.

relative record data set (RRDS). A type of VSAM data set whose records have fixed or variable lengths, and are accessed by relative record number.

remote procedure call (RPC). RPC is an independent set of functions used for accessing remote nodes on a network. Using RPC network services, applications can be created in much the same way a programmer writes software for a single computer using local procedure calls. The RPC protocols extend the concept of local procedure calls across the network, which means that distributed applications can be developed for transparent execution across a network.

Resource Access Control Facility (RACF). An IBM licensed program that is included in z/OS Security Server and is also available as a separate program for the z/OS and VM environments. RACF provides access control by identifying and verifying the users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

Resource Measurement Facility (RMF). An IBM licensed program or optional element of z/OS, that measures selected areas of system activity and presents the data collected in the format of printed reports, system management facilities (SMF) records, or display reports. Use RMF to evaluate system performance and identify reasons for performance problems.

rmdir. An AIX, UNIX, OS/2 and DOS command used to remove (delete) a directory.

RMF. See *Resource Management Facility (RMF)*.

root. The user name for the system user with the most authority.

RPC. Remote procedure call.

RRDS. Relative record data set.

S

SAF. System Authorization Facility (SAF).

SDSF. System Display and Search Facility (SDSF).

sequential file. A type of MVS file that has its records stored and retrieved according to their physical order within the file. It must be on a direct access volume.

server. (1) A functional unit that provides shared services to workstations over a network; for example, a file server, a print server, a mail server. (2) On a

network, the computer that contains the data or provides the facilities to be accessed by other computers in the network. (3) A program that handles protocol, queuing, routing, and other tasks necessary for data transfer between devices in a computer system.

sharing. A term used in a computing environment to refer to utilizing a file on a remote system. It is done by mounting the remote file system, then reading or writing files in that remote system.

shell prompt. The character string on an AIX or UNIX command line indicating the system can accept a command (typically the \$ character).

showattr. A client command used to display the values of the site, processing, and data set creation attributes.

SMF. See *system management facilities (SMF)*.

SMP/E. See *System Modification Program/Extended (SMP/E)*.

SMS. See *Storage Management Subsystem (SMS)*.

SPF. System productivity facility.

SRM. System Resources Manager (SRM).

SSF. Software support facility.

stale file handle. A file handle is stale when the file handle for a file or prefix is no longer valid.

stateless. A stateless server can function correctly without maintaining any protocol state information about any of its clients. The NFS protocol is intended to be as stateless as possible. This avoids complex crash recovery; a client just resends requests until a response is received. The stateless protocol is chosen to minimize the probability of data losses due to a server crash.

storage hierarchy. An arrangement of storage devices with different speeds and capacities. The levels of the storage hierarchy include main storage (memory, DASD cache), primary storage (DASD containing uncompressed data), migration level 1 (DASD containing data in a space-saving format), and migration level 2 (tape cartridges containing data in a space-saving format). See also *primary storage, migration level 1, migration level 2, object storage hierarchy*.

Storage Management Subsystem (SMS). A DFSMS facility used to automate and centralize the management of storage. Using SMS, a storage administrator describes data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements to the system through data class, storage class, management class, storage group, and ACS routine definitions.

system management facilities (SMF). A component of z/OS that collects input/output (I/O) statistics, provided at the data set and storage class levels, which helps you monitor the performance of the direct access storage subsystem.

superuser. The user who can operate without the restrictions designed to prevent data loss or damage to the system (User ID 0).

System Modification Program/Extended. Basic tool for installing software changes in programming systems. It controls these changes at the element (module or macro) level, which helps protect system integrity.

T

tape volume. Storage space on a tape, identified by a volume label, which contains data sets or objects and available free space. A tape volume is the recording space on a single tape cartridge or reel. See also *volume*.

task control block (TCB). An MVS control block that describes an asynchronous process that is called a task.

TCB. See *task control block (TCB)*.

TCP/IP. See *Transmission Control Protocol/Internet Protocol (TCP/IP)*.

TIOT. Task input/output table.

transmission control block (TCB). An internal control block within the TCP/IP address space.

Transmission Control Protocol (TCP). A stream communication protocol that includes error recovery and flow control.

Transmission Control Protocol/Internet Protocol (TCP/IP). The two fundamental protocols of the Internet protocol suite. The abbreviation TCP/IP is frequently used to refer to this protocol suite. TCP/IP provides for the reliable transfer of data, while IP transmits the data through the network in the form of datagrams. Users can send mail, transfer files across the network, or execute commands on other systems.

TSO. Time-sharing option.

TSO/E. Time-sharing option extended.

U

UCB. Unit control block.

UDP. User datagram protocol.

UID. The DFSMSHsm authorized-user identification in 1 to 7 characters. See also *user number (UID)*.

Unicode Standard. A universal character encoding standard that supports the interchange, processing, and display of text that is written in any of the languages of the modern world. It can also support many classical and historical texts and is continually being expanded. The Unicode Standard is compatible with ISO/IEC 10646.

UNIX. A highly portable operating system originally developed by Bell Laboratories that features multiprogramming in a multi-user environment. UNIX is implemented in the C language. UNIX was originally developed for use on minicomputers but has been adapted on mainframes and microcomputers. It is especially suitable for multiprocessor, graphics, and vector-processing systems.

UNIX authentication. Requires a client process to send credentials (the client's machine-name, uid, gid, and group-access-list) to the server.

user catalog. An optional catalog used in the same way as the master catalog and pointed to by the master catalog. It lessens the contention for the master catalog and facilitates volume portability.

User Datagram Protocol (UDP). A connectionless datagram protocol that requires minimal overhead, but does not guarantee delivery.

user number (UID). In the AIX operating system, a number that uniquely identifies a user to the system. It is the internal number associated with a user ID.

V

virtual storage access method (VSAM). An access method for direct or sequential processing of fixed and variable-length records on direct access storage devices. The records in a VSAM file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the file (entry sequence), or by relative record number.

volume. The storage space on DASD, tape, or optical devices, which is identified by a volume label. See also *DASD volume*, *optical volume*, *tape volume*.

VRA. Variable Recording Area (VRA).

VSAM. See *virtual storage access method (VSAM)*.

VSCR. Virtual storage constraint relief.

X

XDR. See *External Data Representation (XDR)*.

Z

z/OS. z/OS is a network computing-ready, integrated operating system consisting of more than 50 base elements and integrated optional features delivered as a configured, tested system. See also *MVS/ESA SP*.

z/OS Network File System. A base element of z/OS, that allows remote access to z/OS host processor data from workstations, personal computers, or any other system on a TCP/IP network that is using client software for the Network File System protocol.

z/OS UNIX System Services file system. In the z/OS UNIX environment, the collection of files and file management structures on a physical or logical mass storage device, such as a diskette, minidisk, or a disk drive. See also *hierarchical file system (HFS) data set*.

z/OS UNIX System Services (z/OS UNIX). The set of functions provided by the SHELL and UTILITIES, kernel, debugger, file system, C/C++ Run-Time Library, Language Environment, and other elements of the z/OS operating system that allow users to write and run application programs that conform to UNIX standards.

Index

Special Characters

(<PROCNAME>) 19

Numerics

0037, EBCDIC code page 29

0C4 ABEND 93

0C4 protection exception 66

A

ABEND

B37 121

D37 121

E37 121

X'0F3' 101

X'806' 101

X'A03' 101

X'x13' 101

X'x22' 101

X'x37' 101

X'x3E' 101

ABEND0C4 93

ABENDxxx keyword 93

access method services error message GFSA847I 119

access modes 85

access to remote files 21

acdirmax attribute 61

acdirmax keyword 138

acdirmin attribute 61

acdirmin keyword 138

acregmax attribute 61

acregmax keyword 138

acregmin attribute 61

acregmin keyword 138

active clients 123

active data sets, displaying 71

AIX, downloading client commands 32

alias 146

alias name, message GFSA817I 117

allocate command 21

allocating

attributes data set 23

checklist data set 26

exports data set 24

mount handle data sets 27

altsym keyword 24

altsym parameter, start command 67

ARC0075E DFSMSHsm message 105

ASCII to EBCDIC conversion 47

Assembler header macro, reference 215

async processing attribute 50

asynchronous daemon task 142

atime 185, 187

attention messages, collecting 73

AttrCaching attribute 61

attribute

security(exports) 11, 13

security(none) 13

security(saf) 11, 177

security(safexp) 12, 177

xlat 29

attribute statement syntax 24

attributes

attributes 12

attrtimeout 50, 109

bufhigh 105, 107, 140

cln_ccsid(n) 52

data set 4

data set creation 44

dsntype 110

dsorg 108

exports 5

for z/OS UNIX System Services 43

logicalcache 107

maxtimeout 109, 110

mintimeout 109, 110

nfstasks 111

noattrtimeout 110

nomaxtimeout 110

noreadtimeout 110

nowritetimeout 110

processing 46

readtimeout 50, 109

recfm 110

safexp 5

security 5, 77

site 54

srv_ccsid(n) 52

text 111

trusted 10

vol 108

writetimeout 50, 109

attributes data set 23

attrtimeout attribute 50, 109

attrtimeout processing attribute 46

authentication

protocols 13

authentication error 118

authority, trusted 10

Authorized Program Analysis Report) 91

authorizing file operations 13

automount facility 22, 23

B

B37 ABEND 121

basic sequential access method (BSAM) 1, 21, 62

binary processing attribute 47

Biod attribute 61

blankstrip 146

blankstrip processing attribute 47

blks data set attribute 44

blksize data set attribute 44

BPXVCLNY load module 66
BSAM (basic sequential access method) 21
bufhigh attribute 105, 107, 140
Bufhigh attribute 61
bufhigh site attribute 55

C

C function, fopen 139
C function, text1 140
C header macro, reference 209
cachewindow site attribute 55
cancel command 68
cancel mvsnfsc command 65
cataloged data sets 3
changing
 attributes 23
 mount handle data sets 72
channel status word, message GFSA876I 121
Character Data Representation Architecture (CDRA) 14, 141
checking UNIX permission bits 58
checklist attribute 12
checklist data set 6
checklist site attribute 55
client, z/OS NFS introduction 1
client attributes 61, 63
client commands, installing 31
client log daemon 141
client log data set, setting up 95
client messages 128
client program requirements 185
client-server relationship 1
client transport handle 127
clients, supported 6
cln_ccsid 52, 112, 138
cln_ccsid attribute 61
cln_ccsid(n) attribute 52
cln_ccsid(n) processing attribute 47
cln_ccsid parameter 8, 14, 63
cln_ccsid setting 21
clnt_call 127
clnt_pcreateerror 126
clnt_perror message 127
clntudp_create 126
closing a data set 72
code
 return code mapping 150
 return codes 149
Coded Character Set Identifier (CCSID) 8, 14, 52, 54, 61, 63, 112, 138
collecting
 diagnostic messages 73
 usage statistics 19
command
 allocate 21
 cancel 68
 cancel mvsnfsc 65
 delim
 data conversion 21
 dir 147

command (*continued*)
 force 65
 ls (UNIX) 47, 48, 182
 make (AIX or UNIX) 34
 make (ULTRIX) 36
 make example 175
 modify 24, 69, 111, 112, 114, 115, 122
 mount 3, 12, 135
 changing default and mount point attributes 44
 mvsllogin 11, 22, 79, 125, 127, 148
 mvsllogin (client) 31
 mvsllogout 79, 125
 mvsllogout (client) 31
 mvsllogout 79
 mvsllogout (client) 31
 rm (UNIX) 48
 rmdir 119
 showattr 125, 127
 showattr (client) 31
 start 24, 67
 stop 67
 stop mvsnfsc 65
 touch (AIX or UNIX) 34
 tso mount 14, 21
 umount (client) 23
 unmount 123, 135
comment symbol 67
component identification keyword 92
configuring the Network File System 23, 27
configuring the NFS 9
configuring the server
 attributes data set 24
 exports data set 24
 mount handle data sets 24
console message (<PROCNAME>) 19
control files 4
controlling access to data sets 24
CONVXLAT utility 29
creating conventional MVS data sets 4
creation (data set creation) attributes 44
credentials, UNIX-style 13
credentials allocation failure, message GFSA827I 118
ctime 185
customizing exit routines 75
customizing the translation table 29
cyls data set attribute 44

D

d mms,all command 17
D37 ABEND 121
data conversion
 delim 21
 filedata 21
 mvs2os2 14
 os22mvs 14
data management error message 121
data set 4
 access 4
 attribute defaults 4
 attributes 4

- data set (*continued*)
 - cataloged 3
 - creating 4
 - default attributes 4
 - definition 3
 - export 5
 - location 4
 - MVS 87
 - organizations supported 3
 - storage 4
 - structure 4
- data set creation attributes 44, 46
- DataCaching 138
- DataCaching attribute 62, 63
- dataclas attribute 44
- DCB (data control block) parameters 44
- DD statements
 - exports 24
 - FHDBASE 27
 - FHDBASE2 27
 - NFSATTR 23
 - NFSCMSG1 22
 - NFSCMSG2 22
 - NFSLOG1 30
 - NFSLOG2 30
 - NFSXLAT 29
- DDNAME EXPORTS 120
- DDNAME NFSATTR 107
- deallocating a data set 72
- debugging, collecting messages for 73
- DelayWrite attribute 62
- DelayWrite keyword 138
- deleting
 - entries from mount handle data sets 72
 - migrated files 49
 - tasks 111
- Delim attribute 62
- delim keyword 139
- delim parameter 21
- development toolkits 38, 40
- DFSMSshsm 49
- diagnosing problems 73
- diagnosis 91
- dir command 147
- dir data set attribute 44
- direct access (DA) 182, 185
- displaying
 - mount points and active data sets 71
 - status of active subtasks 72
- DOC SCnnnnnnnn keyword 94
- Domain Name Server 66, 117
- DOS client, Network File System 31
- double byte character set (DBCS) 14, 28
- downloading
 - commands to the client 31
 - source code to the client 175
- dsntype attribute 110
- dsntype data set attribute 44
- dsorg attribute 108
- dsorg data set attribute 44
- Dump Analysis and Elimination (DAE) 101

- dump data set, setting up 98
- dynamic allocation failure 118, 119
- DynamicSizeAdj attribute 62

E

- E37 ABEND 121
- EBCDIC code page 0037 29
- EBCDIC to ASCII conversion 47
- Electronic Technical Response (ETR) 98
- end-of-line (EOL) terminator 122
- end-of-line specifiers 47
- entry-sequenced data set (ESDS) 21
- envvar parameter 17
- error messages 103
- error messages, collecting 73
- ESDS (entry-sequenced data set) 21
- ESTAE exit routine 112
- estae macro 142
- etc/passwd 127
- etc/passwd file 125
- executebitoff processing attribute 47
- executebiton processing attribute 47
- exit routines 75
- export spanning pathnames 7
- Export Spanning Pathnames 111
- exportfs operand 24, 70, 124
- exporting a file system 12
- exports data set 5, 148
- exports list and SAF checking 11, 12
- External Data Representation (XDR) 2
- extlink processing attribute 48

F

- fastfilesize processing attribute 48, 182
- FAT file system 38
- file creation attributes 44
- File exists message 145
- file security 127
- file security exit 83, 85
 - request code 87
 - return code 87
- file size determination 181
- file system 3
- file timeout 114
- File Transfer Protocol (FTP) 31
- File Usage Block 121
- file usage block, message GFSA881I 121
- filedata parameter 21
- fileextmap processing attribute 48
- files, source for client enabling commands 38
- filesystype parameter 140
- filesystype parmlib statement 20
- fixed-blocked file format 43
- flushing partial records, message GFSA811I 115
- flushlog operand 73
- fn_delimiter site attribute 55
- force command 65
- freeze=off operand 122
- freeze=on operand 122

freeze operand 70
French EBCDIC format (CCSID 297) 14

G

getattr 110
getattr operation 46
GFSA960I 85
GFSA961I 85
GFSASSMF macro 209
GFSAUDSA user storage block 75
GFS AUSMF macro 215
global exit block (GXB) 83
Global Exit Block (GXB) 76, 80
granting access to data sets 24

H

HDZ11TC 92
HDZ11TS 66, 92
HFS() keyword 57
hfs(prefix) site attribute 56
hfsbtimeout site attribute 56
hierarchical file system (HFS) 2, 52, 111, 114, 115
high-level qualifier (HLQ) 3

I

IBMLink/Service, diagnosis using 91
IDCAMS error 119
incorrou keyword 94
informational messages, collecting 73
Input Error 91
installation, planning for 9
installation exits 13
installing the mvslogin, mvslogout, and showattr
commands 31, 175
integrated catalog facility (ICF) 3
Integrated Catalog Facility (ICF) 3
inter-process communication (IPC) queue 113
ISO 8859-(ASCII) 29

J

job control language (JCL) 44, 120

K

Kerberos 136
key-sequenced data set (KSDS) 106
keys data set attribute 45
keyword 91
altsym 24

L

leadswitch site attribute 56
limiting access to data sets 24
list=dsnames operand 123, 124
list=mounts operand 115, 123, 124

list operand 71
Lockd/Statd 17
log data set 5
log data set, setting up 95
log data sets
NFSCMSG1 96
NFSCMSG2 96
NFSLOG1 69, 73, 97
NFSLOG2 69, 73, 97
log=msglevel operand 73
logical file system (LFS) 20
logicalcache attribute 107
logicalcache site attribute 56
login and logout with PCNFSD 177
login exit routine 77
logon and file security 127
logout
logout, forced 77
logout site attribute 56
lookup operation 46
lookup request 7, 111
loop keyword 94
lrecl data set attribute 45
ls (UNIX) command 47, 48, 182

M

make (AIX or UNIX) command 34
make (ULTRIX) command 36
make command example 175
mapleaddot processing attribute 48
maplower processing attribute 48
mapped processing attribute 48
maxfileproc parameter 143
maxrdfsleft site attribute 56
maxsockets 114
maxtimeout attribute 109, 110
maxtimeout site attribute 56
message, console 19
message log data set, setting up 95
messages
A FILE SYSTEM WITH THE SAME NAME IS
ALREADY MOUNTED. 135
CANNOT CREATE UDP SERVICE. 113
CANNOT OPEN CLIENT LOG DATA SET, text1,
text2. 139
Cannot open input file, text1:text2 message 144
Cannot open output file, text1:text2 message 144
CANNOT OPEN THE ATTRIBUTE DATA SET. 107
CANNOT OPEN THE CHKLIST DATA SET. 121
CANNOT OPEN THE EXPORTS DATA SET. 120
Cannot read input file, text1:text2 message 144
Cannot resolve local host name message 144
Cannot write output file, text1: text2 message 144
CATALOG (text) COULD NOT BE LOCATED. 120
CLIENT LOG DATA SET, text, FLUSHED. 139
CLIENT LOG DATA SET SWITCHED TO text. 140
CLIENT LOG DATA SET text RE-INITIALIZED. 139
CLIENT LOGGING ENDED. 139
CLN_CCSID AND SRV_CCSID WILL BE IGNORED
AS XLAT OPTION IS OFF. 138

messages (continued)

COMMAND text NOT VALID. 125
CREATE FAILED FOR text. 119
CREATED TASK(h_digits) - text1 - text2. 111
DATA SET text1 CREATION USING DATA CLASS =
text2. 117
DELETING TASK(h_digits) - text. 111
Directory text not mounted. message 145
DYNAMIC ALLOCATION: INPUT VALIDATION
ROUTINE REJECTED ALLOCATION. 119
Enter MVS password: 127
Enter new MVS password: 127
EOL SEQUENCE MISMATCH FOR DATA SET
text1(text2). 122
Error: Can't open text for read. 127
Error: Directory text not mounted. 127
Error: Drive text not mounted. 127
Error: filesystem text is local. 127
ERROR ENCOUNTERED WHILE PARSING
HOSTNAME, REASON CODE reasoncd. 139
ERROR ENCOUNTERED WHILE PARSING MOUNT
PATH, REASON CODE reasoncd. 138
ERROR IN READING TRANSLATION TABLE,
text. 128
ERROR RETURNED TO CLIENT: RC = d_digits
<text>. 107
ERROR WAS DETECTED IN THE EXPORTS FILE.
EXPORT LIST NOT REBUILT. 124
EXPORT LIST HAS BEEN REBUILT
SUCCESSFULLY. 124
EXPORT SPANNING PATHNAMES NOT
SUPPORTED 111
EXPORTS: DIRECTORY text WAS NOT
EXPORTED. 120
EXPORTS: NO VALID HOST NAMES IN text
LIST. 117
EXPORTS: text1 CANNOT BE EXPORTED
BECAUSE text2 ALREADY IS. 120
EXPORTS: UNEXPECTED OPTION (text)--
SHUTDOWN SCHEDULED. 120
EXPORTS: UNKNOWN HOST (text) HAS BEEN
IGNORED. 120
HFS CANNOT RESOLVE PATH NAME text. 115
HFS MOUNT PROCESSING ACTIVE. 123
HFS MOUNTS RESUMED. 115
HFS MOUNTS SUSPENDED. 114
HFS PROCESSING DISABLED. 123
HFS PUBLIC PATHNAME SPECIFIED BUT HFS IS
NOT ENABLED 111
Host Error: text. 127
Host text1 returned error d_digits: text2 126
IDCAMS ERROR: text. 119
Input path name cannot be equal to output path
name. 144
INSTALLATION DEFAULT TRANSLATION TABLE
CANNOT BE INITIALIZED. 128
INVALID DSNTYPE SPECIFICATION IN LINE
d_digits. 110
INVALID DSORG SPECIFICATION IN LINE
d_digits. 108

messages (continued)

INVALID OPTION SPECIFICATION OF text IN LINE
d_digits. 111
INVALID PUBLIC SPECIFICATION IN LINE
d_digits 111
INVALID RECFM SPECIFICATION (text). 107
INVALID RECFM(text) - MUST SPECIFY U, F, OR
V. 110
INVALID SECURITY SPECIFICATION IN
d_digits 111
INVALID SIDEFILE SPECIFICATION IN LINE
d_digits 111
INVALID SMS_keyword SPECIFICATION IN LINE
d_digits. 110
INVALID SPECIFICATION RECFM(text). 110
INVALID UNIT SPECIFICATION IN LINE
d_digits. 108
INVALID VOL SPECIFICATION IN LINE
d_digits. 108
LOG DATA SET IS RE-INITIALIZED. 125
LOG DATA SET IS SWITCHED FROM text1 TO
text2. 125
LOG DATA SET text IS FLUSHED. 125
MISSING LEFT PARENTHESIS IN LINE
d_digits. 108
MISSING LEFT PARENTHESIS IN text
KEYWORD. 137
MISSING RIGHT PARENTHESIS IN LINE
d_digits. 108
MISSING RIGHT PARENTHESIS IN text
KEYWORD. 137
MOUNT FAILED: text 115
MOUNT HANDLE DATABASE CANNOT BE
READ. 105
MOUNT PROCESSING ACTIVE. 122
MOUNT PROCESSING RESUMED. 122
MOUNT PROCESSING SUSPENDED. 122
Must be a root user to issue 'character' flag
message 144
NETWORK FILE SYSTEM CLIENT DAEMON TASK
RESTARTED. 142
NETWORK FILE SYSTEM CLIENT LOGGING IS
TERMINATED. 141
NETWORK FILE SYSTEM CLIENT SHUTDOWN
COMPLETE. 140
NETWORK FILE SYSTEM CLIENT SHUTDOWN IN
PROGRESS. 140
NETWORK FILE SYSTEM IS SHORT ON
STORAGE. 107
NETWORK FILE SYSTEM SERVER ESTAE EXIT
UNABLE TO COMPLETE PROCESSING.
ABEND=abend_code. 112
NETWORK FILE SYSTEM SERVER
INITIALIZATION FAILED: text 104
NETWORK FILE SYSTEM SERVER LOG text SET
TO FORCELOG. 125
NETWORK FILE SYSTEM SERVER LOGGING IS
TERMINATED. 125
NETWORK FILE SYSTEM SERVER SVC DUMP
REQUEST FAILED. REASON=reason_code. 112
NETWORK SERVICE IS UNAVAILABLE. 113

messages (continued)

NEW AND OLD FILES ARE NOT MEMBERS OF THE SAME PDS. 119
NFSLOG1 OR NFSLOG2 DD STATEMENTS NOT DEFINED. 125
NO ACTIVE DATA SETS. 124
NO ACTIVE HFS DATA SETS. 115
NO ACTIVE HFS MOUNT POINTS. 115
NO ACTIVE MOUNT POINTS. 124
NO SWAP REQUEST FOR NETWORK FILE SYSTEM SERVER FAILED. 105
OPEN FAILED RC d_digits FOR DATA SET text1(text2). 119
OPENEDITION V_REG FAILED: RV=1, RC=h_digit1, RSN=h_digit2. 104
OS/390 NETWORK FILE SYSTEM LOCK MANAGER (fmid) STARTED 106
OS/390 NETWORK FILE SYSTEM LOCK MANAGER SHUTDOWN COMPLETE 106
OS/390 NETWORK FILE SYSTEM STATUS MONITOR (fmid) STARTED 106
OS/390 NETWORK FILE SYSTEM STATUS MONITOR SHUTDOWN COMPLETE 106
OS/390 NFS SERVER CANNOT OBTAIN NFS PORT 2049. 114
OS/390 UNIX REGISTRATION SUCCESSFUL. 114
OS/390 version NETWORK FILE SYSTEM CLIENTfmid STARTED 140
OS/390 version NETWORK FILE SYSTEM SERVER fmid STARTED. 107
PARSE FAILED FOR ATTRIBUTE FIELD - ILLEGAL KEYWORD IN LINE d_digits. 108
PARSE FAILED IN LINE d_digits text. 108
PARSE FAILED ON ALPHABETIC FIELD FOR text KEYWORD. 137
PARSE FAILED ON NUMBER FIELD IN LINE d_digits. 108
PARSE FAILED ON NUMERIC FIELD FOR text KEYWORD. 137
Password change required by host. 126
PDS text IS NOT EMPTY. 119
PROBLEMS ENCOUNTERED PARSING THE CHKLIST DATA SET. 121
PUBLIC PATH CANNOT BE ESTABLISHED. 118
READ FAILED FOR NETWORK FILE SYSTEM CLIENT MOUNT PARAMETERS. 139
READ FAILED FOR THE ATTRIBUTES DATA SET. 108
READ FAILED RC d_digits FOR DATA SET text1(text2). 119
REaddir ON ROOT IS NOT ALLOWED. 120
RECALL FAILED FOR MIGRATED DATA SET text. 105
REGISTER FOR PORT d_digits1 PROGRAM d_digits2 - VERSION d_digits 3 - FAILED. 113
REMOVE FAILED: RC h_digits DSN text1(text2). 116
REQUEST h_digits - FILE text NOT ALLOCATED. 122
REQUEST h_digits CREDENTIALS ALLOCATION FAILED. 118

messages (continued)

REQUEST h_digits INVALID CREDENTIALS FLAVOR d_digits. 118
REQUEST h_digits INVALID MEMBERNAME FOR text. 118
REQUEST h_digits PARSE FAILED FOR text. 118
REQUEST HEADER DATA BLOCK ALLOCATION FAILED. 113
REQUESTED MEMORY NOT AVAILABLE. 105
RETRANS OPTION WILL BE IGNORED AS HARD OPTION IS ON. 138
Retype new MVS password: 127
Retyped password does not match 126
SERVER SHUTDOWN COMPLETE. 105
SERVER SHUTDOWN IN PROGRESS. 105
SERVICE REQUESTER DOES NOT HAVE SECURITY PRIVILEGE. 115
SMF PROCESSING ACTIVE FOR FILE TIMEOUT. 114
SMF PROCESSING ACTIVE FOR USER LOGOUT. 114
SMF PROCESSING SUSPENDED FOR FILE TIMEOUT. 114
SMF PROCESSING SUSPENDED FOR USER LOGOUT. 114
SUBTASK TERMINATED: h_digits. 112
TASK h_digits1 TCB h_digits2 PROGRAM = text1 = text2. 122
text. 107, 123
text : can't find name for uid d_digits. 125
text: Error: cannot determine server. 126
text: Error: unknown return from usage routine. 126
text ACTIVE = d_digits. 123
text DEALLOCATED. 123
text IS AN INCORRECT KEYWORD FOR MOUNT PARAMETER. 138
text IS NOT A VALID DATA SET NAME. 123
text IS NOT A VALID MEMBER NAME. 124
text logged in ok. 126
text NOT ALLOCATED. 123
text NOT MOUNTED. 123
text UNMOUNTED. 123
text1: 127
text1: Error: text2 mounted from server text3, not text4. 126
text1: host "text2" unknown. 126
text1: text2 126, 127
text1(d_digits1) IS SET TO THE DEFAULT VALUE, text1(d_digits2,d_digits3,d_digits4). 111
text1 FAILED FOR text2, text3. 140
text1(text2) 123
text1(text2) ACTIVE = d_digits. 123
text1 UNSUPPORTED DSORG text2. 118
THE ATTRIBUTE VALUE d_digits1 IS NOT IN THE RANGE OF d_digits2 TO d_digits3. 109
THE FILE SIZE IS TOO LARGE. FN=text, OFFSET=h_digit1, LEN=h_digit2. 106
THE IP ADDRESS OF REMOTE HOST NAME, hostname, COULD NOT BE RESOLVED. 136
time_stamp. 107
TRANSLATION TABLE text IS LOADED. 128

messages (continued)

uid text logged out ok. 126
UNABLE TO BIND A RESERVED PORT TO
SOCKET socketnum. 142
UNABLE TO CREATE IPC QUEUE. 113
UNABLE TO SETUP ERROR RECOVERY (ESTAE),
RETURN CODE returncd. 142
UNEXPECTED END OF STRING ON END OF
PARSE IN LINE d_digits. 108
UNEXPECTED ERROR DETECTED: d_digits
text. 107
Unknown flag '-character' 125
Unknown flag '-character' message 144
Unknown host text message 144
UNKNOWN KEYWORD ENCOUNTERED AROUND
POSITION d_digit. 138
UNMOUNT COMMAND FAILED: MOUNT POINT
STILL IN USE. 123
UNREGISTER PROGRAM d_digits1 VERSION
d_digits2 - FAILED. 113
usage: text [-pn] [-g group][-a account] hostname
[mvs_username] 126
usage: text[-a] [-d] [-e] [host] 143
usage: text hostname 126
usage: text input output message 144
usage: text1 [-t] hostname [text2] 127
VERIFY: (text) IS NOT A VSAM DATA SET. 124
VERIFY FAILED WITH RC = d_digits FOR
(text). 124
VERIFY SUCCESSFUL FOR (text). 124
WRITE FAILED RC d_digits FOR DATA SET
text1(text2). 120

messages, client

GFSC100E 134
GFSC101E 134
GFSC102E 134
GFSC103E 134
GFSC105E 134
GFSC106E 134
GFSC107E 134
GFSC110E 135
GFSC200E 135
GFSC201E 135
GFSC202E 135
GFSC203E 135
GFSC204E 135
GFSC205E 135
GFSC206E 135
GFSC207E 136
GFSC208E 136
GFSC209E 136
GFSC210E 136
GFSC211E 136
GFSC212E 136
GFSC213E 136
GFSC214E 137
GFSC300E 137
GFSC301E 137
GFSC302E 137
GFSC303E 137
GFSC304E 137

messages, client (continued)

GFSC305E 137
GFSC307E 138
GFSC308E 138
GFSC309E 138
GFSC310I 138
GFSC311I 138
GFSC312I 138
GFSC313I 138
GFSC315E 138
GFSC317E 139
GFSC318E 139
GFSC319E 139
GFSC320E 139
GFSC500I 139
GFSC501I 139
GFSC502E 139
GFSC503E 139
GFSC504I 140
GFSC505E 140
GFSC506E 140
GFSC700I 140
GFSC701I 140
GFSC702I 140
GFSC703E 140
GFSC704E 140
GFSC705E 140
GFSC707E 141
GFSC708E 141
GFSC709E 141
GFSC710E 141
GFSC711E 141
GFSC712E 141
GFSC713E 141
GFSC714E 142
GFSC715E 142
GFSC716I 142
GFSC721E 142
GFSC722E 142
GFSC723E 142
GFSC724E 142
GFSC725E 143
GFSC726E 143
GFSC727W 143
GFSC728E 143
GFSC729W 143
GFSC840I 143
GFSC841E 144
GFSC843E 144
GFSC845I 144
GFSC846E 144
GFSC847E 144
GFSC848E 144
GFSC849E 144
GFSC850E 144
GFSC854I 144
GFSC855E 144
GFSC856E 145
GFSC858E 145

messages, client platform (AIX)

Cross device link message 145

messages, client platform (AIX) (continued)

Directory Not Empty message 145
File Name Too Long message 145
I/O Error (with possible system programmer response) 145
I/O Error (with possible user response) 146
Is a directory 146
No space left on device 147
No such device 147
No such file or directory 147
Not a directory 147
Not Owner 147
Permission denied 147
Read Only File System 148
Stale NFS File Handle 148
Weak Authorization 148

messages, console

CANNOT CREATE UDP SERVICE. 113
CANNOT OPEN THE ATTRIBUTE DATA SET. 107
CANNOT OPEN THE CHKLIST DATA SET. 121
CANNOT OPEN THE EXPORTS DATA SET. 120
ERROR WAS DETECTED IN THE EXPORTS FILE.
EXPORT LIST NOT REBUILT. 124
EXPORTS: DIRECTORY text WAS NOT EXPORTED. 120
EXPORTS: NO VALID HOST NAMES IN text LIST. 117
EXPORTS: text1 CANNOT BE EXPORTED BECAUSE text2 ALREADY IS. 120
EXPORTS: UNEXPECTED OPTION (text)-- SHUTDOWN SCHEDULED. 120
EXPORTS: UNKNOWN HOST (text) HAS BEEN IGNORED. 120
HFS MOUNT PROCESSING ACTIVE. 123
HFS MOUNTS RESUMED. 115
HFS MOUNTS SUSPENDED. 114
HFS PROCESSING DISABLED. 123
INSTALLATION DEFAULT TRANSLATION TABLE CANNOT BE INITIALIZED. 128
LOG DATA SET IS RE-INITIALIZED. 125
LOG DATA SET IS SWITCHED FROM text1 TO text2. 125
LOG DATA SET text IS FLUSHED. 125
MOUNT HANDLE DATABASE CANNOT BE READ. 105
MOUNT PROCESSING ACTIVE. 122
MOUNT PROCESSING RESUMED. 122
MOUNT PROCESSING SUSPENDED. 122
NETWORK FILE SYSTEM SERVER ESTAE EXIT UNABLE TO COMPLETE PROCESSING. ABEND=abend_code. 112
NETWORK FILE SYSTEM SERVER INITIALIZATION FAILED: text 104
NETWORK FILE SYSTEM SERVER LOG text SET TO FORCELOG. 125
NETWORK FILE SYSTEM SERVER LOGGING IS TERMINATED. 125
NETWORK FILE SYSTEM SERVER SVC DUMP REQUEST FAILED. REASON=reason_code. 112
NETWORK SERVICE IS UNAVAILABLE. 113

messages, console (continued)

NFSLOG1 OR NFSLOG2 DD STATEMENTS NOT DEFINED. 125
NO ACTIVE DATA SETS. 124
NO ACTIVE HFS DATA SETS. 115
NO ACTIVE HFS MOUNT POINTS. 115
NO ACTIVE MOUNT POINTS. 124
NO SWAP REQUEST FOR NETWORK FILE SYSTEM SERVER FAILED. 105
OPENEDITION V_REG FAILED: RV=1, RC=h_digit1, RSN=h_digit2. 104
OS/390 NFS SERVER CANNOT OBTAIN NFS PORT 2049. 114
OS/390 version NETWORK FILE SYSTEM SERVER fmid STARTED. 107
PARSE FAILED IN LINE d_digits text. 108
PROBLEMS ENCOUNTERED PARSING THE CHKLIST DATA SET. 121
PUBLIC PATH CANNOT BE ESTABLISHED. 118
READ FAILED FOR THE ATTRIBUTES DATA SET. 108
RECALL FAILED FOR MIGRATED DATA SET text. 105
REGISTER FOR PORT d_digits1 PROGRAM d_digits2 - VERSION d_digits 3 - FAILED. 113
SERVER SHUTDOWN COMPLETE. 105
SERVER SHUTDOWN IN PROGRESS. 105
SMF PROCESSING ACTIVE FOR FILE TIMEOUT. 114
SMF PROCESSING ACTIVE FOR USER LOGOUT. 114
SMF PROCESSING SUSPENDED FOR FILE TIMEOUT. 114
SMF PROCESSING SUSPENDED FOR USER LOGOUT. 114
TASK h_digits1 TCB h_digits2 PROGRAM = text1 = text2. 122
text. 123
text ACTIVE = d_digits. 123
text DEALLOCATED. 123
text IS NOT A VALID DATA SET NAME. 123
text IS NOT A VALID MEMBER NAME. 124
text NOT ALLOCATED. 123
text NOT MOUNTED. 123
text UNMOUNTED. 123
text1(d_digits1) IS SET TO THE DEFAULT VALUE, text1(d_digits2,d_digits3,d_digits4). 111
text1(text2) 123
text1(text2) ACTIVE = d_digits. 123
UNABLE TO CREATE IPC QUEUE. 113
UNMOUNT COMMAND FAILED: MOUNT POINT STILL IN USE. 123
UNREGISTER PROGRAM d_digits1 VERSION d_digits2 - FAILED. 113
VERIFY: (text) IS NOT A VSAM DATA SET. 124
VERIFY FAILED WITH RC = d_digits FOR (text). 124
VERIFY SUCCESSFUL FOR (text). 124

messages, server

GFS320I(<PROCNAME>) 104
GFS322I(<PROCNAME>) 104

messages, server *(continued)*

GFS3A323I 104
 GFS3A324I 104
 GFS3A325I 105
 GFS3A328I 67
 GFS3A328I(<PROCNAME>) 105
 GFS3A329I 68
 GFS3A329I(<PROCNAME>) 105
 GFS3A330I 68
 GFS3A330I(<PROCNAME>) 105
 GFS3A331E(<PROCNAME>) 105
 GFS3A333I(<PROCNAME>) 105
 GFS3A334I(<PROCNAME>) 105
 GFS3A335E(<PROCNAME>) 105
 GFS3A335I 106
 GFS3A336E(<PROCNAME>) 106
 GFS3A336I 106
 GFS3A337I 106
 GFS3A338I 106
 GFS3A344I 69, 106
 GFS3A345I 68, 106
 GFS3A346I 107
 GFS3A347I 107
 GFS3A348I 67
 GFS3A348I(<PROCNAME>) 107
 GFS3A349I 107
 GFS3A360I 107
 GFS3A361I 107
 GFS3A362I(<PROCNAME>) 107
 GFS3A400I 107
 GFS3A401I(<PROCNAME>) 107
 GFS3A402I(<PROCNAME>) 108
 GFS3A403I 109
 GFS3A403I(<PROCNAME>) 108
 GFS3A404I 108
 GFS3A405I 108
 GFS3A406I 108
 GFS3A407I 108
 GFS3A408I 108
 GFS3A409I 108
 GFS3A411I 108
 GFS3A412I 108
 GFS3A413I 109
 GFS3A414I 109
 GFS3A415I 109
 GFS3A416I 109
 GFS3A417I 109
 GFS3A418I 109
 GFS3A419I 109
 GFS3A420I 109
 GFS3A421I 110
 GFS3A422I 110
 GFS3A423I 110
 GFS3A424I 110
 GFS3A425I 110
 GFS3A426I 110
 GFS3A429I 110
 GFS3A430I 110
 GFS3A431I 52, 111
 GFS3A434I(<PROCNAME>) 111
 GFS3A435E 111

messages, server *(continued)*

GFS4A436I 111
 GFS4A437I 111
 GFS4A438I 111
 GFS4A439I 111
 GFS4A440I 111
 GFS4A450I 111
 GFS4A451I 111
 GFS4A452I 112
 GFS4A470I 101
 GFS4A470I(<PROCNAME>) 112
 GFS4A471I 101
 GFS4A471I(<PROCNAME>) 112
 GFS4A480I 112
 GFS4A481I 112
 GFS4A482I 112
 GFS4A483I 112
 GFS4A501I 113
 GFS4A502I 113
 GFS4A554I(<PROCNAME>) 113
 GFS4A555I 113
 GFS4A556I(<PROCNAME>) 113
 GFS4A557I 113
 GFS4A558I(<PROCNAME>) 113
 GFS4A559I(<PROCNAME>) 113
 GFS4A564I(<PROCNAME>) 113
 GFS4A565I(<PROCNAME>) 114
 GFS4A566I(<PROCNAME>) 114
 GFS4A750I(<PROCNAME>) 114
 GFS4A751I 72
 GFS4A751I(<PROCNAME>) 114
 GFS4A752I(<PROCNAME>) 114
 GFS4A753I 72
 GFS4A753I(<PROCNAME>) 114
 GFS4A754I(<PROCNAME>) 114
 GFS4A770I 114
 GFS4A771I 71
 GFS4A771I(<PROCNAME>) 114
 GFS4A772I 71
 GFS4A772I(<PROCNAME>) 115
 GFS4A776I 115
 GFS4A777I 115
 GFS4A781I 72
 GFS4A782I(<PROCNAME>) 115
 GFS4A783I(<PROCNAME>) 115
 GFS4A784I 115
 GFS4A786I 115
 GFS4A801I 115
 GFS4A802E 116
 GFS4A811I 116
 GFS4A812I 116
 GFS4A814I 116
 GFS4A815I 116
 GFS4A816I 117
 GFS4A817I 117
 GFS4A818I(<PROCNAME>) 117
 GFS4A819I 117
 GFS4A820I 117
 GFS4A821I 117
 GFS4A822I 118
 GFS4A823I(<PROCNAME>) 118

messages, server (continued)

GFSA827I 118
 GFSA829I 118
 GFSA832I 118
 GFSA833I 118
 GFSA840I 118
 GFSA842I 118
 GFSA843I 119
 GFSA847I 119, 124
 GFSA848I 119
 GFSA849I 119
 GFSA853I 118, 119
 GFSA854I 118, 119
 GFSA858I 119
 GFSA859I 119
 GFSA860I 120
 GFSA862I 120
 GFSA863I 120
 GFSA864I(<PROCNAME>) 120
 GFSA865I 93
 GFSA865I(<PROCNAME>) 120
 GFSA866I(<PROCNAME>) 120
 GFSA867I(<PROCNAME>) 120
 GFSA868I(<PROCNAME>) 120
 GFSA869I(<PROCNAME>) 120
 GFSA871I 121
 GFSA876I 121
 GFSA877I 121
 GFSA878I(<PROCNAME>) 121
 GFSA879I(<PROCNAME>) 121
 GFSA881I 121
 GFSA886I 121
 GFSA895I 122
 GFSA896I 122
 GFSA897I 122
 GFSA898I 122
 GFSA899I 122
 GFSA900I 72
 GFSA900I(<PROCNAME>) 122
 GFSA901I 71
 GFSA901I(<PROCNAME>) 122
 GFSA902I 71
 GFSA902I(<PROCNAME>) 122
 GFSA903I 72
 GFSA903I(<PROCNAME>) 122
 GFSA904I(<PROCNAME>) 123
 GFSA908I(<PROCNAME>) 123
 GFSA909E(<PROCNAME>) 123
 GFSA910I 71
 GFSA910I(<PROCNAME>) 123
 GFSA911I(<PROCNAME>) 123
 GFSA912I 72
 GFSA912I(<PROCNAME>) 123
 GFSA913I(<PROCNAME>) 123
 GFSA914I 72
 GFSA914I(<PROCNAME>) 123
 GFSA915I 72
 GFSA915I(<PROCNAME>) 123
 GFSA916I 73
 GFSA916I(<PROCNAME>) 123
 GFSA917I 73

messages, server (continued)

GFSA917I(<PROCNAME>) 123
 GFSA918I(<PROCNAME>) 123
 GFSA919I(<PROCNAME>) 124
 GFSA920I(<PROCNAME>) 124
 GFSA921I(<PROCNAME>) 124
 GFSA922I(<PROCNAME>) 124
 GFSA923I(<PROCNAME>) 124
 GFSA924I(<PROCNAME>) 124
 GFSA925I(<PROCNAME>) 124
 GFSA926I(<PROCNAME>) 124
 GFSA927I(<PROCNAME>) 124
 GFSA930I(<PROCNAME>) 125
 GFSA931I(<PROCNAME>) 125
 GFSA932I(<PROCNAME>) 125
 GFSA933I(<PROCNAME>) 125
 GFSA934I(<PROCNAME>) 125
 GFSA935I(<PROCNAME>) 125
 GFSA936I(<PROCNAME>) 125
 GFSA949I 125
 GFSA950I 125
 GFSA951I 125
 GFSA952I 126
 GFSA953I 126
 GFSA954I 126
 GFSA955I 126
 GFSA956I 126
 GFSA957I 126
 GFSA958I 126
 GFSA959I 126
 GFSA960I 79, 126
 GFSA961I 79, 126
 GFSA964I 126
 GFSA965I 126
 GFSA966I 126
 GFSA967I 127
 GFSA968I 127
 GFSA969I 127
 GFSA970I 127
 GFSA971I 127
 GFSA972I 127
 GFSA973A 127
 GFSA974A 126, 127
 GFSA975A 127
 GFSA975I 126
 GFSA976I 127
 GFSA977I 127
 GFSA978I 127
 GFSA991E(<PROCNAME>) 127
 GFSA996E(<PROCNAME>) 128
 GFSA997I 128
 GFSA998I 128
 text1 REQUEST NOT VALID ON ALIAS NAME
 text2. 117
 mgmtclas data set attribute 45
 migrated files 49
 mintimeout attribute 109, 110
 mintimeout site attribute 57
 modify command 19, 24, 69, 98, 111, 112, 114, 115,
 122
 modify command, operands of 67

- modifying
 - attributes 23
 - checklist data set 26
 - mount handle data sets 72
- mount, umount, login, logout 177
- mount and umount with PCNFSD 177
- mount command 3, 12, 135
- mount handle data set 5, 27
- Mount Handle Database failed 116
- mount point
 - definition 3
- mount processing options 64
- mount TAG option 53, 54
- mounted file space 123
- MSGGFS865I 93
- MSGGFSshnnt keyword 93
- mtime 185, 187
- MTU (Packet Size) 101
- multiple byte character set (MBCS) 28
- multiple data set creation attributes, specifying 43
- MVS file attributes 43
- MVS files
 - attributes 61
- mvslockd procedure 69
- mvslogin 9
- mvslogin command 11, 22, 31, 125, 127, 148, 177
- mvslogout 9
- mvslogout command 31, 125
- mvslogut 9
- mvslogut command 31, 177
- MVSNFS startup procedure 27

N

- namesrv 66
- Native ASCII environment support 52
- native ASCII support 8
- native path 7
- network file system (NFS) 99
- Network File System Client command, text, failed, return value -1 return code returncd reason code reasoncd message 145
- Network File System Protocol Specification, RFC 1094 134
- NFS clients, non-390 based 53
- NFS protocol 6
- NFS servers, non-390 based 53
- nfsattr 147
- NFSCMSG1 log data set 96
- NFSCMSG2 log data set 96
- NFSERR_STALE error response 115
- NFSLOG1 log data set 69, 73, 97
- NFSLOG2 log data set 69, 73, 97
- nfsstat -m 145
- NFSTARB files, downloading 31
- nfstasks attribute 111
- nfstasks site attribute 57
- NFSXLAT DD statement 128
- no space condition 125
- noattrtimeout attribute 110
- noattrtimeout processing attribute 46

- noblankstrip processing attribute 47
- nochecklist site attribute 55
- nofastfilesize processing attribute 48, 183
- nofileextmap processing attribute 48
- nohfs site attribute 56
- noleadswitch site attribute 56
- nomapleaddot processing attribute 48
- nomaplower processing attribute 48
- nomaxtimeout attribute 110
- nomaxtimeout site attribute 56
- nonspanned data set attribute 46
- nopcfsd site attribute 57
- noreadtimeout attribute 110
- noreadtimeout processing attribute 49
- noretrieve processing attribute 49
- norlse data set attribute 45
- nowritetimeout attribute 110
- nowritetimeout processing attribute 50
- number of active clients 123

O

- OEMVS311 translation table 29
- OMVS segment 177
- operands of the MODIFY command 67
- operating the client 65
 - stopping 65
- operating the NFS NLM 68
 - stopping 69
- operating the NFS NSM 68
 - stopping 69
- operating the server 66
 - stopping 67
- overview of Network File System 1

P

- parameter
 - delim 21
 - envar 17
 - filedata 21
 - vers(x) 6
 - xlat 21
- parameter list contents 79
- parameters
 - region 105
 - xlat() 112
- parsing failure in attributes data set, message GFSA403I 108
- partitioned data set (PDS) 4, 29, 182, 186, 187
- partitioned data set extended (PDSE) 4, 29, 44, 186, 187
- password (MVS) 11
- PC-NFS, downloading commands 36
- PCNFSD
 - authentication 9
 - login and logout 177
 - mount and umount 177
 - version 1 protocol 178
- PCNFSD authentication 177
- pcnfsd site attribute 57

- PCNFSD version 2 protocol 178
- percentsteal site attribute 57
- perfm keyword 95
- permission bits, checking UNIX 58
- permissions, security 83
- physical sequential 4
- physical sequential (PS) 185
- planning for installation, security measures 9
- port registration failure, message
 - GFSA554I(<PROCNAME>) 113
- portable operating system interface (POSIX) 1
- porting commands for clients 37
- Portmap 65, 66
- Portmapper 7, 113
- portmapper (port 111), NFS (port 2049) 102
- processing attributes 46, 51
- PROCLIB updates 22
- protecting your data 9
- proto, mount request 136
- proto attribute 7, 62
- protocol
 - PCNFSD version 1 178
 - PCNFSD version 2 178
- PTFs, searching for 98
- public keyword 7, 111, 118
- public(legacy_path,hfs_path) 57

Q

- QSAM (queued sequential access method) 21
- qualifiers, high-level 3
- queued sequential access method (QSAM) 1, 21, 62

R

- R1TC 92
- R1TC for FMID HDZ11TC 93
- R1TS 92
- R1TS for FMID HDZ11TS 93
- RACF (resource access control facility) 9, 11
- RACROUTE, remote client 10
- rdrverf processing attribute 48
- ReadAhead attribute 62
- ReadAhead keyword 138
- readaheadmax site attribute 58
- readdir on root 120
- readdirtimeout processing attribute 58
- readtimeout attribute 50, 109
- readtimeout processing attribute 49
- reason code
 - rsnc 134
- reason code reasoncd 142
- reasoncd reason code 142
- recalling migrated files 49
- recfm attribute 110
- recfm data set attribute 45
- recno, write 121
- record type-42 (subtype 7, subtype 8) 19
- recordsize data set attribute 45
- recs data set attribute 44

- reference
 - Assembler header macro 215
 - C header macro 209
- region parameter 105
- region size 140
- release=dsname(member) operand 123
- release level 92
- release operand 72, 123
- remote procedure call (RPC) 2, 62, 63, 65, 113, 127, 134, 143, 148
- removing entries from mount handle data sets 72
- rename failure, messages GFSA814I and GFSA815I 116
- request code
 - file security exit 87
 - login exit 87
- resize attribute 63
- Resource Access Control Facility (RACF) 10, 115, 140, 147
- restimeout site attribute 5, 58
- restricting access to data sets 24
- resuming mount processing 70
- RETAIN search symptom strings 99
- retc return code 134
- retrans attribute 62
- retrans option 63
- retrieve(nowait) processing attribute 49
- retrieve processing attribute 49
- retrieve(wait) processing attribute 49
- retrieving client commands 32
- retry attribute 63
- return code
 - file security exit 87
 - login exit 87
 - mapping 150
 - returncd 134
 - rsnc 141
- return code 132 122
- return code retc 134
- return code returncd 134, 135, 142
- return codes 149
- returncd return code 134, 135, 142
- RFC 2055 7
- rlse data set attribute 45
- rm (UNIX) command 48
- rmdir command 119
- root directory 2
- RPC 1094 134
- RS/6000, downloading client commands 32
- rsnc reason code 134
- rsnc return code 141

S

- SAFEXP 6
- samples
 - exports data set 167
 - SMF report source code 189
 - startup procedures z/OS NFS client 171
 - startup procedures z/OS NFS NLN 173
 - startup procedures z/OS NFS NSM 172

- samples (*continued*)
 - startup procedures z/OS NFS server 169
- sdump macro 112
- sdump service 101
- searching the IBM database for APARs and PTFs 98
- security 11
 - security attribute 77
 - Security Authorization Facility (SAF) 77
 - security exit 85
 - security(exports) attribute 13
 - security keyword 111
 - security(none) attribute 13
 - security(saf) 12
 - security(saf) attribute 177
 - security(safexp) 12
 - security(safexp) attribute 177
 - security site attribute 58
- server, z/OS NFS
 - control files 4
 - creating conventional MVS data sets 4
- server, z/OS NFS introduction 1
- server log data set, setting up 95
- server messages 103
- service level keyword 95
- set mms-01 command 16
- setownernobody processing attribute 49
- setownerroot processing attribute 49
- setting time stamps 187
- sflmax site attribute 59
- shareoptions 4
 - shareoptions data set attribute 45
- showattr 9
 - showattr client command, installing 31
 - showattr command 125, 127
- shutdown of the client 65
- shutdown of the server 67
- sidefile processing attribute 49
- site attributes 54, 61
- size, file size determination 181
- SMF (System Management Facilities) 19
 - smf=on 114
 - smf operand 74
 - SMF report source code, sample 189
 - smf site attribute 59
 - smfwtm macro 19
 - SMS_keyword 110
 - soft|hard attribute 63
 - Solaris, downloading client commands 32
 - source code, downloading to the client 175
 - space data set attribute 46
 - spanned data set attribute 46
 - specifying
 - Network File System attributes 23
 - the exports data set 24
 - srv_ccsid 52, 138
 - srv_ccsid attribute 63
 - srv_ccsid(n) attribute 52
 - srv_ccsid(n) processing attribute 50
 - srv_ccsid parameter 8, 14, 63
 - srv_ccsid setting 21
 - stacks, TCP/IP 18
 - stale NFS file handle 73
 - start command 24, 67, 68
 - starting the client 65
 - starting the NFS NSM, NLM 68
 - starting the server 18, 66
 - statistics, SMF 19
 - status operand 72, 114, 122, 123
 - STEPLIB DD statement 105
 - stop command 67
 - stop mvsnfsc command 65
 - stop operand 68
 - stopping subtasks 111
 - stopping the client 65
 - stopping the server 67
 - storage block 76, 85
 - storage block sample, GFSAUDSA 75
 - storclas attribute 46
 - stow macro 116
 - submitting an APAR 99
 - subtype 7 114
 - subtype 7, record type-42 19
 - subtype 8 114
 - subtype 8, record type-42 19
 - SunOS, downloading client commands 32
 - supported clients 6
 - suspending mount processing 70
 - switchlog 125
 - symbolic links 7
 - SYNAD error, message GFSA877I 121
 - sync processing attribute 50
 - syntax diagrams, how to read xvi
 - SYSERR DD statement 95
 - System Authorization Facility (SAF) 6, 11, 12, 13, 27, 115
 - system-managed facility (SMF) 114
 - System Management Facilities (SMF) 19
 - system toolkits 39, 40
 - SYSUDUMP DD statement 98

T

- tag option 138
- TAR utility 31
- task input/output table (TIOT) 145
- TCB address 111
- TCP/IP, stacks 18
 - tcpip.ETC.RPC, modifying 30
- text attribute 111
- text processing attribute 47
- time_increment 186
- time stamps 185
- timeo attribute 63
- timeout attributes, interaction
 - attrtimeout attribute, interaction 50
 - readtimeout attribute, interaction 50
 - writetimeout attribute, interaction 50
- toolkits
 - development 38, 40
 - system 39, 40
- touch (AIX or UNIX) command 34
- trace, collecting messages for 73

- translation table, for z/OS UNIX 29
- translation tables 28
- Transmission control Protocol/Internet Protocol (TCP/IP) 127
- Transmission Control Protocol/Internet Protocol (TCP/IP) 1, 66, 101, 127, 134
- trks data set attribute 44
- trusted attribute 10
- tso mount command 14, 21
- tso unmount 116

U

- ULTRIX, downloading client commands 35
- umount client command 23
- unexpected return code, SMF 114
- Unicode 64
- Unicode Conversion Services 52
- Unicode Services 14, 17, 112, 141
- unit data set attribute 46
- UNIX kernel 141
- UNIX permission bits, checking 11, 58
- UNIX-style credentials 13
- UNIX System Services
 - OpenEdition MVS 115, 127, 134, 137
- unmount command 123, 135
- unmount operand 72, 123
- USA ASCII format (CCSID 437) 14
- usage statistics, collecting 19
- user datagram protocol (UDP) 113, 143
- user exit block (UXB) 77
- User Exit Block (UXB) 80
- user storage block, login exit 76
- using keywords for diagnosis 91
- UW12345 95
- UW12345 service level keyword 92

V

- validation of MVS passwords and user IDs 11
- Variable Recording Area (VRA) 101
- verify operand 124
- vers, mount request 136
- vers(3) parameter 136
- vers(x) parameter 6
- VFS_MOUNT operation 136
- VFS_PFSCTL operation 136
- VFS_RECOVER operation 136
- VFS_STATFS operation 135
- VFS_SYNC operation 135
- virtual storage access method (VSAM) 1, 4, 21, 62, 124, 185, 186
- virtual storage access method (VSAM) 106
- vol attribute 108
- vol data set attribute 46
- volume data set attribute 46
- VSAM 183
- VSAM (virtual storage access method) 21
- VSAM data sets
 - dsorg attribute 44
 - recordsize data set attribute 45

- VSAM data sets (*continued*)
 - recs data set attribute 44
 - shareoptions data set attribute 45
 - spanned data set attribute 46
- VSAM KSDS data sets, spanned data set attribute 45
- VTAM 101

W

- wait keyword 93
- WebNFS support 7
- writetimeout 147
- writetimeout attribute 50, 109
- writetimeout processing attribute 50
- wsize attribute 63

X

- X'0F3' ABEND 101
- X'806' ABEND 101
- X'A03' ABEND 101
- X'x13' ABEND 101
- X'x22' ABEND 101
- X'x37' ABEND 101
- X'x3E' ABEND 101
- XDR (External Data Representation) 2
- xlat() parameter 112
- xlat attribute 29, 63
- xlat keyword 52
- xlat parameter 14, 21
- xlat processing attribute 50
- xlat(Y) option 138

Z

- z/OS NFS client startup procedures, sample 171
- z/OS NFS NLM startup procedures, sample 173
- z/OS NFS NSM startup procedures, sample 172
- z/OS NFS server startup procedures, sample 169
- z/OS UNIX System Services 1

Readers' Comments — We'd Like to Hear from You

**z/OS
Network File System
Customization and Operation**

Publication No. SC26-7417-02

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>				

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>				
Complete	<input type="checkbox"/>				
Easy to find	<input type="checkbox"/>				
Easy to understand	<input type="checkbox"/>				
Well organized	<input type="checkbox"/>				
Applicable to your tasks	<input type="checkbox"/>				

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? Yes No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name

Address

Company or Organization

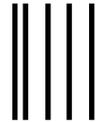
Phone No.



Fold and Tape

Please do not staple

Fold and Tape



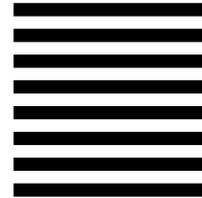
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
RCF Processing Department
M86 050
5600 Cottle Road
San Jose, CA U.S.A. 95193-0001



Fold and Tape

Please do not staple

Fold and Tape



Program Number: 5694-A01



Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC26-7417-02

